

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет имени первого Президента России Б. Н. Ельцина»

УТВЕРЖДАЮ

Директор по образовательной деятельности

М.М.М.М.

С.Т. Князев

« 7 » *сентября* 2023 г.



Прикладные и наукоемкие задачи искусственного интеллекта

Учебно-методические материалы по направлению подготовки
09.03.03 Прикладная информатика
Образовательная программа «Прикладной искусственный интеллект»

Екатеринбург

РАЗРАБОТЧИКИ УЧЕБНО-МЕТОДИЧЕСКИХ МАТЕРИАЛОВ

Доцент Базовой кафедры
«Аналитика больших данных и
методы видеоанализа»

A handwritten signature in blue ink, appearing to read 'Чернавин', is written over a horizontal line.

Н.П. Чернавин

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	5
Глава 1. МАТЕМАТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ И МАШИННОЕ ОБУЧЕНИЕ	8
1.1. Происхождение терминов «математическое программирование» и «машинное обучение»	8
1.2. Основные понятия машинного обучения и математического программирования	12
1.3. Виды признаков	16
1.4. Первоначальная обработка исходных данных	17
1.5. Общая постановка задачи математического программирования	18
Глава 2. ЗАДАЧИ РЕГРЕССИИ И КЛАССИФИКАЦИИ КАК ЗАДАЧИ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ ...	20
2.1. Представление задач регрессии как ЗМП	20
2.2. Задачи регрессии повышенной сложности	30
2.3. Линейно разделимые множества	36
2.4. Метод опорных векторов и потенциальных функций	39
2.5. Оценка качества решения в задачах классификации	42
Глава 3. КОМИТЕТНЫЕ ПОДХОДЫ К РЕШЕНИЮ ЗАДАЧ КЛАССИФИКАЦИИ	49
3.1. Линейно неразделимые множества и метод комитетов	49
3.2. Итерационный алгоритм разделения на основе комитета старшинства	52
3.3. Модель комитета единогласия	63
3.4. Модель комитета большинства	66
3.5. Модель комитета старшинства	70
3.6. Модель комитета с подбором весов	76
3.7. Разделение множеств нелинейными функциями	83
3.8. Применение метода комитетов в ирисах Фишера и оценке информативности признаков	84

Глава 4. ПРИМЕНЕНИЕ ВЫПУКЛЫХ ОБОЛОЧЕК В ЗАДАЧАХ КЛАССИФИКАЦИИ	87
4.1. Одноклассовая классификация	87
4.2. Выпуклые оболочки	88
4.3. Решение одноклассовых задач выпуклыми оболоч- ками	91
4.4. Решение многоклассовых задач выпуклыми оболоч- ками	95
4.5. Взаимосвязь метода комитетов и МВО, эталонные со- стояния	106
Глава 5. ВМЕСТО ПОСЛЕСЛОВИЯ. ФИЛОСОФСКИЕ ПРОБЛЕМЫ МАШИННОГО ОБУЧЕНИЯ	114
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	123

ВВЕДЕНИЕ

В последние годы резко возрос интерес к искусственному интеллекту (ИИ) и машинному обучению (МО). В настоящий момент слова «нейронные сети», «machine learning», «big data», «распознавание образов» в обычном разговоре используют люди всех слоев общества от школьника до пенсионера. Появилось много хороших, готовых для использования программ машинного обучения. К числу таковых, например, можно отнести функции Classify, Predict, ImageIdentify в Wolfram Mathematica 11. Аналогичные функции имеются в MATLAB. На языках высокого уровня, таких как PYTHON или R, уже написано огромное количество программ для решения различных задач машинного обучения. Свои нейронные сети предлагают Яндекс и Google, да и другие IT-компании стараются от них не отставать.

В докладе Еврокомиссии в 2018 г. был констатирован следующий факт: «Как в свое время паровые двигатели и электричество, **искусственный интеллект** меняет мир, общество и промышленность. Вследствие роста вычислительной мощности, доступности данных и прогресса в математических алгоритмах **искусственный интеллект** стал стратегической технологией XXI века».

Несложно предсказать, что в ближайшие десять лет широко-масштабное использование методов ИИ приведет к серьезному изменению бизнес-среды и как следствие к большим социальным потрясениям. По мнению Клауса Шваба, основателя и президента Всемирного экономического форума в Давосе, в ближайшие 20 лет такие профессии, как специалисты по телефонным продажам, по оформлению налоговой документации, страховые оценщики, должностные лица в спортивной индустрии (судьи, арбитры и т.д.), секретари, официанты, агенты по продаже недвижимости, подрядчики в индустрии сельского хозяйства, курьеры и разносчики, могут быть практически полностью автоматизированы [4].

И хотя вероятно, что многие специальности просто окажутся ненужными, но зато будут востребованы другие. Понадобятся специалисты, которые умеют не только использовать готовые средства МО, но и сами их создавать. Более того, умение использовать и создавать программы МО, видимо, в ближайшее время станет таким же необходимым атрибутом современного человека, как знание математического анализа для инженера.

Есть различные взгляды на дальнейшее развитие искусственного интеллекта от предсказания в 2020-е годы очередной «зимы» до продолжения экспоненциального роста интереса к нему и появления «сильного ИИ» в период между 2040 и 2075 гг.

Несмотря на разницу во взглядах на будущее, большинство специалистов сходятся во мнении, что в настоящий момент мы используем только «слабый ИИ» и одним из серьезных препятствий для появления «сильного» является практически полное отсутствие возможности интерпретировать получаемые решения, особенно в задачах классификации. Практических специалистов не устраивает просто ответ, к какому классу отнесен тестируемый объект, им нужна мотивировка, т.е. формализованное решающее правило, которое они могли бы осмыслить и согласиться с ним или нет. Действительно, если речь идет о таких важных вопросах, как сбивать или нет самолет, нарушивший границу, срочно удалять жизненно важный орган или выносить судебный приговор, то для принятия решения хотелось бы иметь четкую аргументацию.

На наш взгляд, интерпретируемость многих задач МО может быть существенно повышена, если их сформулировать как задачи математического программирования (ЗМП). Отметим, что многие чисто математические задачи: сетевого планирования, теории игр и даже оптимального управления представимы как ЗМП. Отсюда не следует, что в таком виде они эффективнее решаются, но воспринимаются и осмысливаются в большинстве случаев точно лучше.

Цели данной монографии состоят в следующем:

- на математических моделях показать, что действительно многие задачи МО являются по сути ЗМП;

- в максимально доступной форме, с использованием геометрических и содержательных интерпретаций научить построению решающих правил для различных задач машинного обучения на основе ЗМП;

- дать практические навыки написания программ в кодах IBM ILOG CPLEX для решения сформулированных задач;

- достичь понимания читателями, что предполагаемые методы универсальны (не путать с единственными) и могут быть использованы в различных видах деятельности (экономика, медицина,

геология, биология, химия, социология, неразрушающий контроль и пр.);

— показать, что приобретенные навыки в дальнейшем могут быть использованы и для решения задач из области исследования операций.

По возможности мы старались соблюдать такую последовательность действий:

- 1) общие рассуждения, как возникает некоторая ситуация;
- 2) конкретный пример в удобном для использования виде;
- 3) геометрическая интерпретация;
- 4) математическая модель;
- 5) результаты расчетов, их геометрическая и смысловая интерпретация, практические области, где эти навыки могут быть применены.

Предлагаемый в данной монографии подход к решению задач МО через сведение их к ЗМП и дальнейшему решению ЗМП стандартными оптимизаторами (CPLEX, MIP, PULP) успешно использовался авторами для решения ряда сугубо практических задач. Мы не будем подробно описывать суть каждой из них, а приведем лишь ссылки на соответствующие статьи.

1. Анализ котировок ценных бумаг и курсов валют на фондовых и валютных биржах [28–30].

2. Оценка технических сигналов и волатильности фондового рынка [29].

3. Оценка кредитного риска по потребительским кредитам [19, 25, 37, 38].

4. Рейтинговые модели оценки заемщиков [21–24, 26, 39–41].

5. Определение латентных больных туберкулезом по анамнезу и анализам крови [43].

6. Регрессионные, имитационные и оптимизационные модели подбора оптимальных параметров агломерационного процесса [30].

Глава 1

МАТЕМАТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ И МАШИННОЕ ОБУЧЕНИЕ

1.1. ПРОИСХОЖДЕНИЕ ТЕРМИНОВ «МАТЕМАТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ» И «МАШИННОЕ ОБУЧЕНИЕ»

Жизнь человека состоит из бесконечной череды принимаемых решений. Даже самые простые из них зависят от ряда факторов или параметров. Например, решение о времени пробуждения определяется состоянием здоровья, рабочий завтра день или выходной, назначены с утра встречи или нет, а если назначены, то сколько необходимо времени туда добраться и т.п. Причем ряд параметров могут быть конкретизированы более точно. В частности, при оценке состояния здоровья можно положиться на самочувствие (хорошо или не очень) или померить температуру и давление, т.е. придать абстрактной оценке конкретное значение и на их основе оценить состояние здоровья и принять решение: принять лекарство, лечь в постель или вызвать скорую.

Оценка параметров и принятие решения происходят на основе собственного опыта или мнения экспертов (родителей, жены, друзей, справочников, рекламы). Причем собственный опыт может быть формализованным (любым способом, например даже методом проб и ошибок) или неформализованным (интуитивным или случайным).

Все последующие действия:

- 1) делать зарядку или нет;
- 2) брать с собой зонтик или нет;
- 3) каким видом транспорта добираться до работы;
- 4) покупать или продавать;
- 5) строить или ломать;
- 6) делать операцию или нет;
- 7) ставить оценку «хорошо» или «плохо» и т.д. —

укладываются в простую схему (рис. 1):

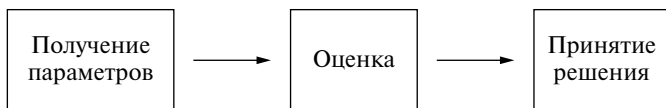


Рис. 1. Схема процесса принятия решения

Кроме того, на процесс принятия решения влияет система ограничений (наличие денег, мнение начальства, собственников, ограничение по химическому составу или физическим параметрам и т.п.) и целевая установка (максимизация прибыли, минимизация рисков, стоимость изделия, сроков строительства и т.п.).

Таким образом, можно утверждать, что процесс выработки решения есть некоторая задача оптимизации целевой установки или установок при системе ограничений. Естественным является стремление к формализации процесса принятия решений, т.е. к записи его в понятной для себя и других форме, причем такой, чтобы существовали методы решения поставленных задач за разумное время и можно было оценить качество полученного решения.

Иными словами, необходимо описать данную задачу как задачу *математического программирования*. Математическое программирование – это раздел математики, разрабатывающий теорию и численные методы решения многомерных задач с ограничениями¹. Для ведения дальнейшего повествования необходимо дать определение основным понятиям в данной области знаний.

В рамках решения задач математического программирования нами изучается множество *объектов исследования*, описываемых совокупностью некоторых входных параметров. Например, при принятии решения конкретным человеком параметрами могли являться совокупность его знаний, опыта и мнения окружающих. В рамках математического программирования входные параметры объекта исследования называют *переменными*.

В свою очередь, для принятия решения необходима некоторая целевая установка, которая позволяет нам сравнивать решения и однозначно говорить, что одно решение лучше или хуже другого. На языке математического программирования такие целевые установки носят название *целевая функция*.

Из множества соотношений между переменными, системой ограничений и целевой функции складывается модель математического программирования².

¹ Методы оптимизации: Учеб. пособие / Н.В. Бразовская; Алтайский государственный технический университет им. И.И. Ползунова [Центр дистанц. обучения]. Барнаул: Изд-во АлтГТУ, 2000, 120 с.

² Гасс С. Путешествие в Страну Линейного Программирования. М.: Мир, 1971. 176 с.

В рамках математического программирования выделяют обширный класс задач *линейного программирования*, в котором, в свою очередь, математическая модель описывается исключительно линейными соотношениями. В дальнейшем в данной работе достаточно часто инструментарий линейного программирования будет использоваться нами для решения задач машинного обучения. Но прежде необходимо определить, что представляет собой термин «машинное обучение».

Машинное обучение – это обширный подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных по мере накопления опыта относительно некоторого класса задач увеличивать качество их решения относительно выбранной целевой функции³.

Конечно, данное выше определение не является единственным. Приведем другие, которые отражают ситуацию в этой области знаний.

Машинное обучение (англ. machine learning, ML) – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а **обучение** в процессе применения решений множества сходных задач⁴.

Машинное обучение (machine learning) – обширный подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных обучаться.

Машинное обучение находится на стыке математической статистики, методов оптимизации и классических математических дисциплин, но имеет также и собственную специфику, связанную с проблемами вычислительной эффективности и переобучения. Многие методы индуктивного обучения разрабатывались как альтернатива классическим статистическим подходам⁵.

«**Машинное обучение** – как секс в старших классах. Все говорят о нем по углам, единицы понимают, а занимается только препод. Статьи о машинном обучении делятся на два типа: это либо трехтомники с формулами и теоремами, которые я ни разу не смог дочитать даже до середины, либо сказки об *искусственном интеллекте, профессиях будущего и волшебных дата-саентистах*»⁶.

Последнее определение, несмотря на солидную долю юмора, достаточно точно отражает ситуацию в области преподавания машинного обучения.

³ Саймон Х. Нейронные сети: полный курс / Х. Саймон. М.: Издательский дом Вильямс, 2006. 1104 с.

⁴ https://ru.wikipedia.org/wiki/Машинное_обучение

⁵ https://habr.com/ru/hub/machine_learning/

⁶ https://vas3k.ru/blog/machine_learning/

Вообще термин «машинное обучение» — не совсем удачный перевод с английского на русский, из которого неспециалисту трудно понять, человек обучает машину или машина человека. Более точным является следующее определение:

- В то время когда многие пытались разгадать коды немецкой шифровальной машины «Enigma» вручную, своей головой, английский математик и криптограф Алан Тьюринг создавал машину, которая сама будет разгадывать этот код.
- Machine learning — это **обучение машины** для того, чтобы она решала задачи определенного типа без участия человека⁷.

Однако термин «машинное обучение» устойчиво закрепился в русском языке, поэтому мы тоже будем оперировать им.

Теперь настала пора взглянуть на машинное обучение с гносеологической точки зрения. В гносеологии (теории познания) принято разделять методы познания на индуктивные и дедуктивные.

- **Индукция:** переход от частного знания к общему. Наиболее известным философом, развивающим этот подход, был Фрэнсис Бэкон.
- **Дедукция:** переход от общего знания к частному. Наиболее яркими представителями этого подхода были Аристотель, Рено Декарт, Готфрид Лейбниц.

С гносеологической точки зрения различают два вида обучения:

- **Обучение по прецедентам**, или **индуктивное обучение**, основано на выявлении эмпирических закономерностей в данных.
- **Дедуктивное обучение** предполагает формализацию знаний экспертов и их перенос в компьютер в виде базы знаний (экспертные системы).

Обучение по прецедентам, общая постановка. Имеется множество *объектов* (ситуаций) и множество возможных *ответов* (откликов, реакций). Существует некоторая зависимость между ответами и объектами, но она неизвестна. Известна только конечная совокупность *прецедентов* — пар «объект, ответ», называемая *обучающей выборкой*. На основе этих данных требуется восстановить неявную зависимость, т.е. построить алгоритм, способный для любого возможного входного объекта выдать достаточно точный ответ⁸.

Термины «машинное обучение» и «обучение по прецедентам» можно считать синонимами.

Соответственно **задача машинного обучения** — установить некоторую зависимость, т.е. построить алгоритм, способный для наблюдения выдать достаточно точный ответ.

⁷ https://neural-university.ru/neural-networks-basics/?utm_source=email&utm_content=0808

⁸ https://ru.wikipedia.org/wiki/машинное_обучение

Если формулировать более научно, то задача машинного обучения состоит в нахождении функции (алгоритма), приближающей множество объектов к множеству ответов:

$$F(X) \rightarrow Y,$$

где X – множество объектов, Y – множество ответов.

Основные типы задач машинного обучения:

1. **Обучение с учителем** (классификация, регрессия) – множество *ответов* существует.
2. **Обучение без учителя** (кластеризация, фильтрация, сокращение размерности и иные) – множество *ответов* отсутствует.

Что означает термин «обучить»? Обучить означает подобрать коэффициенты в формуле (формулах) решающего правила.

Решающее правило – это любая совокупность действий, на основании которых по множеству входных признаков объекта исследования можно определить его выходные признаки.

В рамках данной монографии наше внимание будет сосредоточено на задачах регрессии, классификации и связанных с ними задачах информативности и генерации признаков. Поэтому дадим определение этих понятий.

1.2. ОСНОВНЫЕ ПОНЯТИЯ МАШИННОГО ОБУЧЕНИЯ И МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Регрессия (лат. regressio – обратное движение, отход) в теории вероятностей и математической статистике – математическое выражение, отражающее зависимость (линейную или нелинейную) зависимой переменной y от независимых переменных x при условии, что это выражение будет иметь статистическую значимость.

Этот термин в статистике впервые был использован Фрэнсисом Гальтоном (1886) в связи с исследованием вопросов наследования физических характеристик человека. В качестве одной из характеристик был взят рост человека; при этом было обнаружено, что в целом сыновья высоких отцов, что не удивительно, оказались более высокими, чем сыновья отцов с низким ростом. Более интересным было то, что разброс в росте сыновей был меньшим, чем разброс в росте отцов. Так проявлялась тенденция возвращения роста сыновей к среднему (*regression to mediocrity*), т.е. «регресс»⁹.

⁹ [https://ru.wikipedia.org/wiki/Регрессия_\(математика\)](https://ru.wikipedia.org/wiki/Регрессия_(математика))

Классификация – один из разделов машинного обучения, посвященный решению следующей задачи. Имеется множество «объектов», разделенных некоторым образом на «классы». Задано конечное множество объектов, для которых известно, к каким классам они относятся. Классовая принадлежность остальных объектов неизвестна. Требуется получить одну или несколько функций (решающих правил), способных классифицировать произвольный объект из исходного множества. Такие функции называются дискриминантными. Каноническая дискриминантная функция имеет следующее математическое представление¹⁰:

$$d = b + p_1 * x_1 + p_2 * x_2 + \dots + p_n x_n,$$

где d – классифицирующая переменная; p_n – коэффициент дискриминантной функции для n -го объекта; b – свободный член, обеспечивающий выполнение требуемых условий; x_n – дискриминационные переменные для n -го объекта.

В соответствии с каноническим уравнением дискриминантной функции выделяются следующие виды переменных:

1. Классифицирующая (зависимая) переменная, представляющая собой значение класса объекта.

2. Дискриминационные (независимые) переменные, по которым выявляются различия между объектами с разной классовой принадлежностью.

В машинном обучении задачи, при которых нам известна классифицирующая переменная для части исследуемых объектов, относятся к разделу *обучения с учителем*. Стоит отметить, что существуют также задачи обучения без учителя, когда разделение объектов обучающей выборки на классы не задается и требуется классифицировать объекты на основе их сходства друг с другом. В этом случае принято говорить о задачах кластеризации или таксономии и классы называть соответственно кластерами или таксонами¹¹. Однако в рамках данной работы будут рассмотрены только задачи обучения с учителем, т.е. задачи регрессии и классификации.

Выборка – это конечный набор наблюдений, извлеченный из множества всех существующих наблюдений (**генеральной совокупности**). В рамках проведения машинного обучения выделяют два основных вида выборки:

Обучающая выборка – выборка данных, на которой обучается модель.

¹⁰ Ким Дж.О., Мюллер Ч.У. Факторный дискриминантный и кластерный анализ. 1989.

¹¹ Там же.

Контрольная (тестовая) выборка — выборка, по которой оценивается качество построенной модели.

Такое разделение **генеральной** совокупности позволяет избежать проблемы переобученности модели. Модель называется **переобученной**, когда средняя **ошибка** обученного алгоритма на наблюдениях из **контрольной** выборки **значительно выше**, чем на **обучающей** выборке.

Тест — это операция, при которой на основе материала обучения ведется распознавание объектов, ранее не участвовавших при обучении, однако принадлежность их к определенным классам заранее известна. Такой материал обучения называется **тестовой (контрольной) выборкой**.

И в качестве последнего определения рассмотрим понятие *искусственная нейронная сеть*. Если не касаться математической сути данного термина, то его, как правило, определяют как математическую модель, построенную по принципу организации и функционирования биологических нейронных сетей живого организма¹². Являясь по сути лишь одним из методов машинного обучения, искусственная нейронная сеть стала современным синонимом не только всего машинного обучения, но и искусственного интеллекта в целом. Однако, несмотря на реально значимые практические достижения, достигнутые с применением нейронных сетей, не стоит забывать, что по своей математической сути нейронная сеть есть задача математического программирования. Представленные в следующих главах комитетные конструкции будут примером простейшей однослойной нейронной сети.

Теперь, определив основные термины и понятия, начнем изложение материала с наглядных геометрических интерпретаций. Любой объект изучения (человек, ответственная деталь для самолета, месторождение полезных ископаемых, фондовый рынок, состояние погоды и т.п.) может быть охарактеризован конечным набором параметров.

Начнем с человека. Каждый из нас имеет рост и вес. Если рассматривать только эти параметры, то математически любой человек может быть представлен как точка в двухмерном пространстве. При одном и том же росте люди могут иметь разный вес и наоборот. Для полноты картины мы можем наращивать количество параметров. Соответственно добавим возраст — и точка будет в трехмерном пространстве (рис. 2).

¹² Николенко С., Кадурын А., Архангельская Е. Глубокое обучение. 2020.

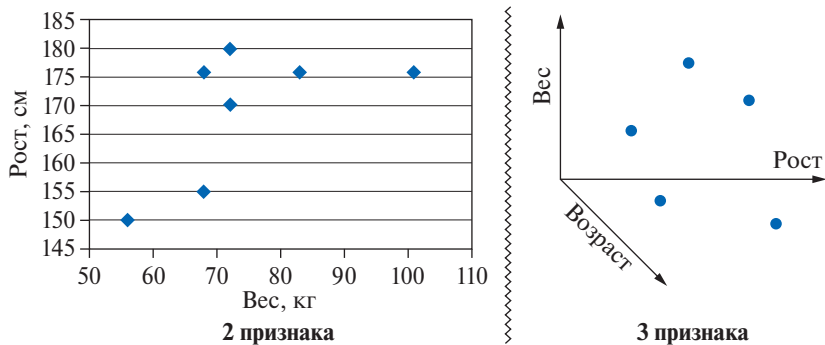


Рис. 2. Отображение человека как точки в пространстве

Однако, если мы попробуем двигаться дальше, например добавить пол и представить точку в четырехмерном пространстве, то уже возникают проблемы с визуализацией, так как пространства размерностью выше трех затруднительно даже мысленно представить (рис. 3).

Иными словами, начиная с 4-мерного пространства понятная геометрическая интерпретация невозможна, но по сути это не имеет значения, так как математические методы не зависят от размерности пространства (числа рассматриваемых параметров). Поэтому попытаемся интерпретировать предлагаемый подход для случая двухмерного пространства, имея при этом в виду, что реальные задачи будут иметь размерности значительно больше двух. Поэтому там, где двухмерной интерпретации недостаточно, будут следовать дополнительные пояснения.

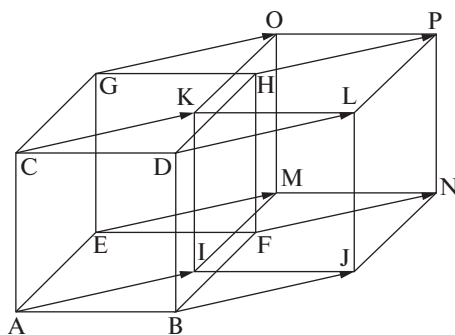


Рис. 3. Пример визуализации 4-мерного пространства¹³

Дальнейший набор параметров будет естественно зависеть от области изучения. Если нам необходимо принять решение (или сделать прогноз) о состоянии здоровья, то целесообразно добавить температуру тела, систолическое/диастолическое давление крови,

¹³ Статья с сайта Habr «Сборка 4-мерного кубика Рубика». [Режим доступа]: <https://habr.com/post/270935/>

возможно, химический состав крови и другие медицинские параметры.

Если нам необходимо принять решение о выдаче человеку кредита, то дополнительными параметрами могут быть: размер заработной платы, наличие в собственности квартиры, машины, дачи, состоит ли в браке, количество детей, частота смен мест работы, сколько раз до этого брал кредит и были ли случаи невозврата кредита или задержки уплаты процентов и другие социальные параметры в зависимости от желаемой глубины исследования и точности принимаемого решения.

После того как определились с набором изучаемых параметров, необходимо сформировать два множества изучаемых объектов: одно для построения решающего правила (обучающая выборка) и другое для проверки решающего правила (контрольная выборка). Кроме того, определяемся, по какому признаку разделяем множества: здоровый/больной, хороший/плохой заемщик, и делим их по этому признаку.

1.3. ВИДЫ ПРИЗНАКОВ

Начнем с человека:

1. Количественные признаки: рост, вес, возраст, годовой доход, температура тела, давление крови. Такие признаки изначально даются в числовом выражении.

2. Качественные признаки:

a. **Бинарные признаки:** пол, наличие квартиры. Данные признаки применяют только значения Да/Нет и кодируются как 1/0 соответственно.

b. **Номинальные признаки:** например, город проживания (Москва, Воронеж, Екатеринбург) — это описательные признаки, по которым нельзя ранжировать данные.

c. **Порядковые признаки:** образование (среднее, высшее, ученая степень и т.д.)

Суть качественных/категориальных признаков состоит в том, что они показывают принадлежность к какой-либо категории (пол, национальность, страна проживания, номер группы, категория товаров и т.п.). Чаще всего данные признаки изначально представлены в нечисловом формате, для решения задач машинного обучения эти признаки кодируются — переводятся в понятный для машины язык.

Существует множество способов кодирования признаков:

1. Бинарная кодировка (binary encoding). Данный способ подходит для кодирования бинарных признаков, принимающих значения Да/Нет и кодировку 0/1.

2. Кодирование метки (label encoding). При такой кодировке каждому признаку присваивается числовое значение, например, Москва = 1, Екатеринбург = 2, Саратов = 3 и т.п.

3. Порядковая кодировка (ordinal encoding). При такой кодировке каждому признаку присваивается числовое значение, при этом учитываются наши знания о ранге, например, отсутствие образования = 0, школьное образование = 1, высшее = 2, ученая степень = 3.

4. Прямое кодирование (one hot encoding). В данном способе кодировки для каждого возможного значения признака вводятся фиктивные переменные, принимающие значения 0/1. Например:

id	type	is_cat	is_dog	is_snake	is_turtle
1	cat	1	0	0	0
2	dog	0	1	0	0
3	snake	0	0	1	0

5. Кодирование по среднему (mean encoding). При такой кодировке каждому значению признака присваиваются средние значения целевой переменной при соответствующем значении признака.

В целом существует множество способов кодировки категориальных признаков, которые показывают разную эффективность в зависимости от задачи.

1.4. ПЕРВОНАЧАЛЬНАЯ ОБРАБОТКА ИСХОДНЫХ ДАННЫХ

Очевидно, что качество исходных данных существенно влияет на получаемые решающие правила. В машинном обучении существует простая поговорка *garbage in – garbage out* (**мусор на входе – мусор на выходе**), и она применима для любых задач. Конечно, качество данных в некотором смысле понятие субъективное и взгляды на него могут изменяться в зависимости от специфики задачи. Однако даже исходя из здравого смысла уже на этой стадии можно дать простые рекомендации, чего следует избегать. Будем различать данные **некачественные по форме** и **некачественные по сути**. Причем это могут быть как **отдельные наблюдения**, так и **признак в целом** для всех наблюдений.

Данные некачественные по форме обычно возникают из-за организационных или технических ошибок при сборе информации. Например, признак должен быть количественным, но у некоторых наблюдений он текстовый или признак должен быть больше нуля,

но попадают отрицательные значения. Естественно, наблюдения с некачественными данными должны быть обнаружены и либо исправлены, либо исключены. Самым простым и надежным, но трудоемким способом является обычный просмотр признаков в Excel и выделение некачественных наблюдений при помощи функции Фильтр.

Данные некачественные по сути можно подразделить исходя из следующих принципов:

1. Несоответствие их физическому, химическому, экономическому и т.д. смыслу, т.е. содержательному смыслу. Коэффициент полезного действия не может быть более 100%; доля коксика в агломерационной шихте не может равняться как нулю, так и единице, а может быть только в каких-то разумных границах; количество забракованных изделий не может быть более 100%. В избавлении от таких признаков лучшие помощники – здравый смысл и эксперты в конкретной области.

2. Бессмысленность присутствия ряда признаков. Например, если есть два признака, значения которых абсолютно одинаковы во всех наблюдениях, то один из них явно лишний, так как неинформативен. Более подробно проблему информативности признаков рассмотрим в отдельной главе.

3. Признаки внешне выглядят нормально, но одновременное присутствие их в задаче создает вычислительные и интерпретационные сложности. Такая ситуация возникает в случае, если признаки линейно зависимы или мультиколлинеарны.

1.5. ОБЩАЯ ПОСТАНОВКА ЗАДАЧИ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Как считал великий математик Эйлер, если тщательно взглянуть в любую задачу, то вы обнаружите, что она на минимум или максимум¹⁴. Действительно, жизнь устроена так, что всегда приходится добиваться некоторой цели, укладываясь при этом в некоторую систему ограничений. Так вот, если вам удалось формализовать функцию цели и систему ограничений, то вы уже сформулировали ЗМП.

Если функция цели и система ограничений линейны, то это задача линейного программирования (ЗЛП), которая имеет очень эффективные алгоритмы решения задач большой размерности. Если это ЗЛП, но ряд переменных целочисленные, то это ЗЛП с частично целочисленными переменными. Причем, если целочисленные

¹⁴ Подробнее см.: <https://ru.citaty.net/avtory/leonard-eiler/>

переменные булевы, то такие задачи, естественно, решаются медленнее, чем чистые ЗЛП, но тоже за приемлемое время.

Наконец, если функция цели или система ограничений нелинейны, то это задачи нелинейного программирования. Данные задачи подразделяются на два класса: выпуклого программирования и невыпуклого.

В классе задач выпуклого программирования есть задачи квадратичного программирования, которые тоже имеют эффективные алгоритмы решения. Так что, если вы практик и смогли сформулировать задачу как ЗЛП или ЗЛП с частично целочисленными переменными или задачу квадратичного программирования, то вы можете воспользоваться алгоритмами их решения, даже не вникая в их суть.

Данные алгоритмы создавались лучшими математиками мира начиная с середины 40-х годов прошлого века. Если у вас есть желание понять и суть самих алгоритмов, вы можете ознакомиться с ними отдельно в рамках курсов по математическому программированию или самостоятельно, читая труды классиков в этой области.

Алгоритмы решения ЗМП оформляются по правилам некоторых программ, которые называются оптимизаторами (CPLEX, MIP, PULP, ...) [50–51]. Для того чтобы ими воспользоваться, необходимо модель и данные привести к воспринимаемому ими виду. Мы будем работать с оптимизатором CPLEX, используя для этого пакет программ IBM ILOG CPLEX. Описание моделей в данном пакете, как, впрочем, и во всех остальных, максимально приближено к обычной математической записи. Поэтому все модели будем записывать первоначально через математические символы, а потом переводить их в коды IBM ILOG CPLEX.

Глава 2

ЗАДАЧИ РЕГРЕССИИ И КЛАССИФИКАЦИИ КАК ЗАДАЧИ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ

2.1. ПРЕДСТАВЛЕНИЕ ЗАДАЧ РЕГРЕССИИ КАК ЗМП

Начнем с более формального определения:

«**Математическое программирование** — это математическая дисциплина, в которой разрабатываются методы отыскания экстремальных значений целевой функции среди множества ее возможных значений, определяемых ограничениями»¹.

Далее, для того чтобы записывать модели в компактной, но понятной всем форме, необходимо вспомнить ряд математических символов.

Сумму математически обозначают заглавной греческой буквой Σ (сигма).

Примеры использования:

Пусть $i = 1, 2, 3$, тогда

$$\sum_{i \in I} p_i = p_1 + p_2 + p_3.$$

Пусть $J_1 = 1, 2$, тогда

$$\sum_{i \in I} p_{ij} < 0 \quad j \in J_1 \rightarrow p_{11} + p_{21} + p_{31} < 0$$

$$p_{12} + p_{22} + p_{32} < 0.$$

Пусть $J_2 = 3, 4, 5$, тогда

$$\sum_{i \in I} p_{ij} > 0 \quad j \in J_2 \rightarrow p_{13} + p_{23} + p_{33} > 0$$

$$p_{14} + p_{24} + p_{34} > 0$$

$$p_{15} + p_{25} + p_{35} > 0.$$

¹ <https://www.intuit.ru/studies/courses/1020/188/lecture/4917>

В данных случаях символы i, j – индексы. Записи $i \in I, j \in J$ читаются как i принадлежит множеству I, j принадлежит множеству J .

Теперь запишем задачу линейной регрессии как ЗМП для двухмерного случая с критерием минимум суммы квадратов невязок (отклонение прогнозного выходного параметра от его фактического значения). Допустим, что множество наблюдений мы обозначаем символом J , а конкретное наблюдение – индексом j . Тогда математически модель будет выглядеть следующим образом:

$$P_j \cdot a + b = Y_j + w_j \quad j \in J \quad (1.1)$$

$$\min \sum_{j \in J} w_j^2. \quad (1.2)$$

Здесь P_j – входной, Y_j – выходной параметр j -го наблюдения; a, b – соответственно искомые коэффициент и свободный член уравнения регрессии; w_j – невязка j -го наблюдения.

Таким образом, условия (1.1) описывают систему ограничений (в данном случае в виде системы уравнений), а условия (1.2) описывают целевую функцию, которую необходимо минимизировать.

Теперь представим данную математическую модель в кодах IBM ILOG CPLEX

```

/* уравнение регрессии двухмерный случай минимум суммы квадратов
невязок */
/*описание данных*/
int K=...; // число элементов в множестве
range j = 1..K; // индекс элемента множества
float P[j]=...; // множество входных параметров объектов
float Y[j]=...; // множество выходных параметров объектов
/* искомые переменные уравнения регрессии */
dvar float a; // коэффициент
dvar float b; // свободный член
dvar float w[j]; // невязка
/* целевая функция*/
minimize sum(j in j) w[j]^2;
/*система ограничений*/
subject to {
forall(j in j) P[j]*a+b==Y[j]+w[j];
};

```

Дадим пояснения:

Комментарии в программе записываются двумя способами либо между `/*текст*/`, либо после `//текст`

int K=...; означает, что в данных некоторой целочисленной константе будет присвоено конкретное значение. Можно было это значение присвоить в тексте программы следующим образом: int K=12;

range j = 1..K; означает, что индекс j изменяется от 1 до K

float P[j]=...; означает, что в данных будет массив P с данными с десятичной точкой. Аналогично Y[j].

dvar float a; означает, что в модели будет переменная «a» непрерывного типа и произвольного знака. Если записать dvar float+a; то переменная может принимать только положительные значения. Если dvar int a; то целочисленные. Если dvar boolean a; то булевы (1 или 0). Аналогично b.

dvar float w[j]; означает, что в модели будет массив переменных w

minimize sum(j in J) w[j]^2; эквивалентна $\min \sum_{j \in J} w_j^2$

subject to {forall(j in J) P[j]*a+b==Y[j]+w[j];}; эквивалентна

$$P_j \cdot a + b = Y_j + w_j \quad j \in J$$

Из приведенного примера видно, что если модель записана корректно математически, то ее достаточно легко перевести в коды IBM ILOG CPLEX.

Теперь необходимо подать на вход модели данные в правильном виде. Сделаем это на примерах:

```
/*случай 1 */
```

```
K=12;
```

```
P=[10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120];
```

```
Y=[6, 10, 17.5, 21, 23.5, 34, 36, 37, 50.5, 51, 61.5, 67];
```

```
/*случай 2 */
```

```
K = 12;
```

```
P = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120];
```

```
Y = [6, 10, 17.5, 21, 23.5, 34, 36, 22, 50.5, 51, 61.5, 67];
```

Данные массива заключаются в квадратные скобки. В качестве разделителя используется запятая или пробел. Естественно, что для каждого случая используется соответствующий ему набор данных.

В данном случае в качестве целевой функции (некоторого критерия качества решения) использовали **минимум суммы квадратов невязок**. Естественно, что могут быть и другие критерии, например **минимум суммы невязок по абсолютной величине**, или необходимо найти такое решение, в котором **максимальная невязка по абсолютной величине должна быть минимальной**. Применение различных критериев обычно связано со спецификой практической задачи и набора данных для ее решения. Поясним это на простых примерах, которые мы уже использовали ранее. Случай 2 отличается от Случая 1 всего одним наблюдением, которое резко отклоняется

от остальных. Причем это может быть достоверное наблюдение и недостоверное. В первом случае его необходимо учесть. Во втором желательно исключить. Для исключения случайных выбросов удобно использовать еще один критерий: **минимизация числа наблюдений, не попавших в заданный диапазон.**

Теперь покажем, как будут изменяться математические модели и коды программ в зависимости от целевой функции. Начнем с **минимума суммы невязок по абсолютной величине.** Если вы используете IBM ILOG CPLEX, то изменения предельно просты. В математической модели целевая функция приобретает вид: $\min \sum_{j \in J} |w_j|$.

В тексте программы соответственно `minimize sum (j in j) abs(w[j]);`

Однако не во всех пакетах математического программирования существует такая возможность. Поэтому лучше использовать классический вид задачи линейного программирования:

$$P_j \cdot a + b = Y_j + w_j - v_j \quad j \in J \quad (1.3)$$

$$w_j \geq 0, v_j \geq 0 \quad j \in J \quad (1.4)$$

$$\min \sum_{j \in J} (w_j + v_j) \quad (1.5)$$

В данном случае w_j и v_j — невязки для отдельного учета положительных и отрицательных отклонений. Так как сами невязки положительны, а их сумма минимизируется, то они не могут одновременно принимать значения больше нуля. Соответственно сделаем изменения в программе.

`/* уравнение регрессии двухмерный случай минимум суммы невязок */`

```
int K=...; // число элементов в множестве
range j = 1..K; // индекс элемента 1-го множества
float P[j]=...; // множество входных параметров объектов
float Y[j]=...; // множество выходных параметров объектов
```

`/* искомые переменные */`

```
dvar float a; // коэффициент
dvar float b; // свободный член
dvar float+ w[j]; // невязка
dvar float+ v[j]; // невязка
```

```
minimize sum(j in j) w[j] + sum(j in j) v[j];
```

```
subject to {
forall(j in j) P[j]*a+b==Y[j]+w[j]-v[j];
};
```

Далее приведем тексты программ для случаев:

1. **Максимальная невязка по абсолютной величине должна быть минимальной.**
2. **Минимум числа наблюдений, не попавших в заданный диапазон.**

Приведем только тексты программ и предложим читателям по ним восстановить и записать самостоятельно математические модели.

```
/* уравнение регрессии двухмерный случай максимальная невязка минимальна */
```

```
int K=...; // число элементов в множестве  
range j = 1..K; // индекс элемента 1-го множества  
float P[j]=...; // множество входных параметров объектов  
float Y[j]=...; // множество выходных параметров объектов
```

```
/* искомые переменные */
```

```
dvar float a; // коэффициент  
dvar float b; // свободный член  
dvar float+ w[j]; // невязка  
dvar float+ v[j]; // невязка  
dvar float+ maxwv; // максимальная невязка
```

```
minimize maxwv;  
subject to {  
forall(j in j) P[j]*a+b==Y[j]+w[j]-v[j];  
forall(j in j) w[j]<=maxwv;  
forall(j in j) v[j]<=maxwv;  
};
```

```
/* уравнение регрессии двухмерный случай минимум числа невязок превышающих заданный диапазон */
```

```
int K=...; // число элементов в множестве  
int L=...; // большое число  
range j = 1..K; // индекс элемента 1-го множества  
float P[j]=...; // множество входных параметров объектов  
float Y[j]=...; // множество выходных параметров объектов
```

```
/* искомые переменные */
```

```
dvar float a; // коэффициент  
dvar float b; // свободный член  
dvar float e[j]; // допустимое отклонение  
dvar float w[j]; // невязка  
dvar boolean z[j]; // индикатор отклонения выше заданного предела  
minimize sum(j in j) z[j];  
subject to {
```



```

forall(j in j) P[j]*a+b==Y[j]+e[j]+w[j];
forall(j in j) -3 <=e[j]<=3;// абсолютное значение
forall(j in j) w[j]<=L*z[j];
forall(j in j) w[j]>= -L*z[j];
};

```

В данном примере мы задали **абсолютное значение допустимого диапазона**. Предлагаем читателю самостоятельно изменить его на **относительное к $Y[j]$** . Один из вариантов решения приведем в параграфе 2.2, посвященном более сложным задачам регрессии.

Приведем результаты решения различных моделей для обоих случаев.

Минимум суммы невязок:

- $y_1 = 0,57 * x_1 - 1,4, y_2 = 0,57 * x_2 - 1,4$

Минимум суммы квадратов невязок:

- $y_1 = 0,5451 * x_1 - 0,84848, y_2 = 0,52937 * x_2 - 1,0758$

Максимальная невязка минимальна:

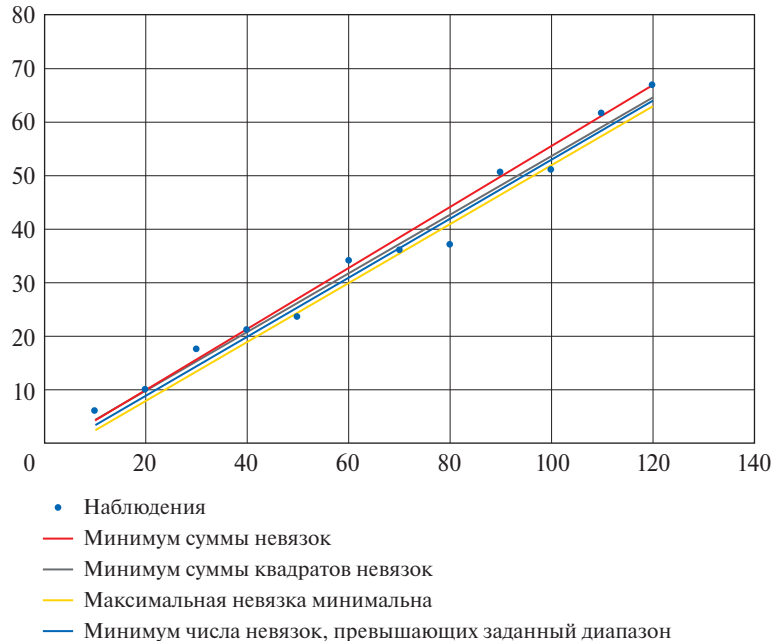
- $y_1 = 0,55 * x_1 - 3 \max wv = 4, y_2 = 0,55 * x_2 - 10,5 \max wv = 11,5$

Минимум числа невязок, превышающих заданный диапазон

- $y_1 = 0,55 * x_1 - 2, y_2 = 0,55 * x_2 - 2$

Графически это будет выглядеть следующим образом:

Случай 1



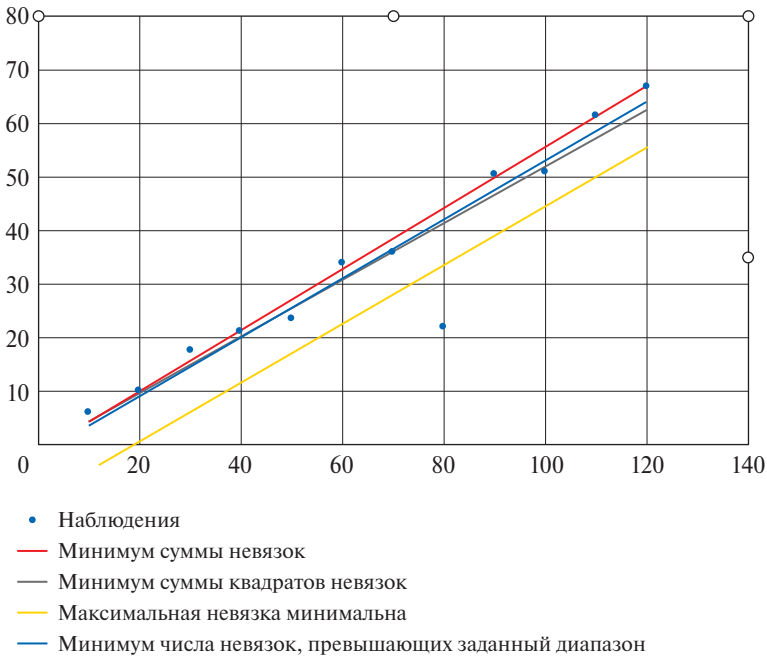
Анализ решений Случай 1

	min-->	суммы невязок		суммы квадратов невязок		максимальной невязки		суммы нарушений диапазона		
x	Y	$y=0.57*x-1.4$	невязка	$y=0.5451*x-0.84848$	невязка	$y=0.55*x-3$	невязка	$y=0.55*x-2$	невязка	
	10	6	4.3	1.7	4.60252	1.39748	2.5	3.5	3.5	2.5
	20	10	10	0	10.05352	-0.0535	8	2	9	1
	30	17.5	15.7	1.8	15.50452	1.99548	13.5	4	14.5	3
	40	21	21.4	-0.4	20.95552	0.04448	19	2	20	1
	50	23.5	27.1	-3.6	26.40652	-2.9065	24.5	-1	25.5	-2
	60	34	32.8	1.2	31.85752	2.14248	30	4	31	3
	70	36	38.5	-2.5	37.30852	-1.3085	35.5	0.5	36.5	-0.5
	80	37	44.2	-7.2	42.75952	-5.7595	41	-4	42	-5
	90	50.5	49.9	0.6	48.21052	2.28948	46.5	4	47.5	3
	100	51	55.6	-4.6	53.66152	-2.6615	52	-1	53	-2
	110	61.5	61.3	0.2	59.11252	2.38748	57.5	4	58.5	3
	120	67	67	0	64.56352	2.43648	63	4	64	3
			минимальная	-7.2		-5.7595		-4		-5
			максимальная	1.8		2.43648		4		3

Анализ решений Случай 2

		min-->		суммы невязок		суммы квадратов невязок		максимальной невязки		сумма нарушений диапазона			
	x	Y	$y=0.57*x-1.4$	невязка	$y=0.52937*x-1.0758$	невязка	$y=0.55*x-10.5$	невязка	$y=0.55*x-2$	невязка			
		10	6	4.3	-1.7	4.2179	-1.7821	-5	-11	3.5	-2.5		
		20	10	10	0	9.5116	-0.4884	0.5	-9.5	9	-1		
		30	17.5	15.7	-1.8	14.8053	-2.6947	6	-11.5	14.5	-3		
		40	21	21.4	0.4	20.099	-0.901	11.5	-9.5	20	-1		
		50	23.5	27.1	3.6	25.3927	1.8927	17	-6.5	25.5	2		
		60	34	32.8	-1.2	30.6864	-3.3136	22.5	-11.5	31	-3		
		70	36	38.5	2.5	35.9801	-0.0199	28	-8	36.5	0.5		
		80	22	44.2	22.2	41.2738	19.2738	33.5	11.5	42	20		
		90	50.5	49.9	-0.6	46.5675	-3.9325	39	-11.5	47.5	-3		
		100	51	55.6	4.6	51.8612	0.8612	44.5	-6.5	53	2		
		110	61.5	61.3	-0.2	57.1549	-4.3451	50	-11.5	58.5	-3		
		120	67	67	0	62.4486	-4.5514	55.5	-11.5	64	-3		
			минимальная		-1.8		-4.5514		-11.5		-3		
			максимальная		22.2		19.2738		11.5		20		

Случай 2



Из приведенных таблиц и рисунков видно, что наиболее чувствительны к случайным выбросам критерии минимум суммы квадратов невязок и минимум максимального абсолютного значения невязки. В случае минимизации числа наблюдений, не попадающих в заданный диапазон, случайный выброс был просто проигнорирован.

Теперь настало время рассмотреть, как изменятся математические модели и программы в многомерном случае, т.е. если входных признаков несколько. Например, необходимо научиться прогнозировать систолическое давление крови пациентов в зависимости от роста, веса и возраста. Естественно, что нам понадобится дополнительный индекс для обозначения этих параметров. Математическая модель в данном случае приобретает следующий вид:

Многомерный случай:

$$\sum_{i \in J} p_{ij} * a_i + b = y_j + w_j \quad j \in J \quad (1.6)$$

$$\min \sum_j w_j^2 \quad j \in J \quad (1.7)$$

А программы соответственно:

```

/* уравнение регрессии многомерный случай минимум суммы квадратов невязок */
int K=...; // число элементов в множестве
range j = 1..K; // индекс элемента 1-го множества
int n=...; // число переменных (параметры наблюдений)
range i = 1..n; // индекс параметра
float P[j][i]=...; // множество входных параметров объектов
float Y[j]=...; // множество выходных параметров объектов
/* искомые переменные */
dvar float a[i]; // коэффициент
dvar float b; // свободный член
dvar float w[j]; // невязка
minimize sum(j in j) (w[j])^2;
subject to {
forall(j in j) sum(i in i) P[j][i]*a[i]+b==Y[j]+w[j];
};

```

Представление данных в многомерном случае будет следующее:

```

/* данные для многомерного случая */
n = 3;
K = 12;
P = [[150, 60, 30],
[180, 75, 40],
[175, 88, 25],
[183, 77, 33],
[172, 75, 45],
[176, 107, 53],
[167, 90, 60],
[170, 80, 65],
[186, 100, 28],
[177, 95, 35],
[165, 70, 25],
[174, 70, 19]];
Y = [130, 125, 140, 122, 124, 170, 150, 130, 129, 135, 128, 120];

```

```

/* уравнение регрессии многомерный случай минимум суммы невязок */
int K=...; // число элементов в множестве
range j = 1..K; // индекс элемента 1-го множества
int n=...; // число переменных (параметры наблюдений)
range i = 1..p; // индекс параметра
float P[j][i]=...; // множество входных параметров объектов
float Y[j]=...; // множество выходных параметров объектов

```

```

/* искомые переменные */
dvar float a[i]; // коэффициент
dvar float b; // свободный член
dvar float+ w[j]; // невязка
dvar float+ v[j]; // невязка
minimize sum (j in j) (w[j]+v[j]);
subject to {
forall (j in j) sum(i in i) P[j][i]*a[i]+b==Y[j]+w[j]-v[j];
};

```

Клише для других целевых функций рекомендуем читателю сделать самостоятельно и апробировать их на многомерных данных.

2.2. ЗАДАЧИ РЕГРЕССИИ ПОВЫШЕННОЙ СЛОЖНОСТИ

Если в задаче регрессии целевая функция строится по методу наименьших квадратов, то такой подход обычно обозначают MSE (Mean Squared Error). Данный подход был предложен и развит Гауссом и Лежандром еще на границе 18-го и 19-го веков [45]. Как ЗМП в компактной форме данный подход может быть записан следующим образом:

$$\min \sum_{j \in J} \left(\sum_{i \in I} p_{ij} * a_i + b - y_j \right)^2.$$

Все обозначения имеют тот же смысл, что и ранее.

Если целевая функция минимум суммы невязок по абсолютной величине, то такой подход обозначают MAE (Mean Absolute Error) и как ЗМП он может быть записан следующим образом:

$$\min \sum_{j \in J} \left| \sum_{i \in I} (p_{ij} * a_i + b - y_j) \right|.$$

Достоинства и недостатки каждого из них мы уже рассмотрели. Естественным образом возникает идея совместить два этих подхода, т.е. целевая функция – MSE (для ошибок ниже порога) и MAE (для ошибок выше порога). В честь автора данного подхода такая регрессия называется регрессией Хубера [48]. Мы думаем, что вам уже не составит труда представить ее как ЗМП.

Вернемся к мультиколлинеарности признаков. В этом случае оценки коэффициентов уравнения регрессии могут быть неустойчивыми, например несоразмерно большими. Простой подход к решению проблемы через исключение сильно коррелирующих

признаков не всегда удобен. Поэтому попробуем совместить в одной модели два критерия. Например, следующим образом:

$$\min\left(\sum_{j \in J} \left(\sum_{i \in I} p_{ij} * a_i + b - y_j\right)^2 + L * \sum_{i \in I} a_i^2 + L * b^2\right),$$

где L – штрафной коэффициент. Этот подход называется гребневая регрессия (Ridge regression) [49]. Очевидно, что регулировать величины a_i и b можно и другим способом:

$$\min\left(\sum_{j \in J} \left(\sum_{i \in I} p_{ij} * a_i + b - y_j\right)^2 + L * \sum_{i \in I} |a_i| + L * |b|\right).$$

Такой метод называется Lasso, а если объединить Lasso и Ridge, то получим подход Elastic net. По сути все три подхода предназначены для одновременной минимизации функции потерь и величины коэффициентов уравнения регрессии. При $L = 0$ гребневая регрессия и метод Lasso будут эквивалентны MSE. По мере увеличения L коэффициенты и свободный член уравнения регрессии будут уменьшаться. При некотором значении L **некоторые коэффициенты a_i станут равными нулю, т.е. мы начнем избавляться от неинформативных признаков.** Процесс увеличения L продолжается до тех пор, пока идет улучшение качества решения.

Гребневая регрессия в основном используется для предотвращения переобучения. Кроме того, она хорошо работает при наличии сильно коррелированных признаков. Регрессию Лассо следует использовать при наличии большого числа признаков. Признаки с нулевыми коэффициентами из дальнейшего рассмотрения можно исключить.

Естественно, что, умея представлять задачи регрессии как ЗМП, вы можете в подходах Ridge и Lasso заменить функцию потерь на МАЕ и получить новые подходы. Более того, можно использовать и другие функции потерь, с которыми мы уже с вами работали. Всеми этими примерами мы хотим показать, что если вы научитесь представлять задачи регрессии как ЗМП, то перед вами открываются широчайшие возможности по решению задач, которые вы не сможете решить стандартными способами. При этом не надо заикливаться на мультиколлинеарности, на практике вам встретятся задачи и посложней. Одну из них мы уже приводили, когда надо случайные выбросы просто проигнорировать, но не известно какие. В компактной форме эту задачу можно записать следующим образом:

$$\min \sum_{j \in J} z_j \tag{1.8}$$

$$(1 - d) * y_j - L * z_j \leq \sum_{i \in I} p_{ij} * a_i + b \leq (1 + d) * y_j + L * z_j$$

$$j \in J, \quad (1.9)$$

где d – допустимый диапазон отклонения от выходных параметров ($0 < d < 1$); z_j – булева переменная (признак попадания или непадения в диапазон d); L – некоторая достаточно большая константа (значительно превышающая y).

С подобного рода задачами мы столкнулись при прогнозировании оборотов торгового эквайринга и прогнозировании качественных характеристик агломерата. Последнюю задачу опишем подробнее, так как без данного подхода нам было бы сложно ее решить.

Имеется задача подбора оптимальных параметров агломерационного процесса. Процесс агломерации относится к сложным, многомерным, растянутым во времени технологическим процессам. Качественные параметры агломерата зависят от большого числа факторов (химико-минералогического и гранулометрического состава сырья, условий увлажнения, дозирования, смешивания и укладки шихты, режимов спекания и охлаждения агломерата). Кроме того, на качественные показатели сильно влияют различного рода шумы от сбоев оборудования и ошибок персонала до резкого изменения погодных условий. Причем различные нарушения могут иметь сходные проявления и в силу сложности и растянутости во времени процесса агломерации своевременно обнаружить и устранить их достаточно сложно.

Заметим, что в рамках исследования была поставлена задача построения именно уравнения, которое бы явно показывало, как изменятся качественные параметры агломерата в зависимости от изменения перечисленных факторов. Поэтому требовалось построить уравнение регрессии, которое будет самостоятельно игнорировать нетипичные наблюдения, поскольку в практических задачах при значительном числе переменных заранее сложно определить, какие наблюдения являются именно выбросами, а какие могут быть объяснены моделью.

Для этого была использована модель, приведенная выше. Всего имелось 2500 наблюдений на обучающей выборке и 700 наблюдений на тестовой выборке, число признаков регрессионной модели 43.

В таблице ниже приведены результаты расчетов модели в сравнении с другими типами регрессии, наш подход назван MILP Regression.

**Результаты модели в сравнении с другими методами регрессии
на тестовой выборке**

	MAPE (средняя абсолютная ошибка)	Процент прогнозов с ошибкой менее 10%
LinearRegression	5,70%	88,36%
Ridge	5,97%	86,68%
Lasso	5,61%	83,03%
ElasticNet	5,65%	84,57%
HuberRegressor	5,84%	83,45%
RandomForestRegressor	5,35%	86,40%
XGB_R	5,76%	87,10%
MILP Regression	5,47%	92,15%

MAPE (Mean Absolute Percentage Error) – средняя абсолютная ошибка в %.

Заметим, что в данном подходе у нас дополнительно возникает информация о том, какие наблюдения были признаны выбросами для регрессионной модели, что также дает дополнительные возможности для дальнейшего анализа конкретных выбросов и определения причин выбросов.

Вообще проблемам робастности (устойчивости к помехам) в машинном обучении уделяется большое внимание. Довольно часто слова *устойчивость* и *робастность* используются как синонимы. На наш взгляд, *устойчивость* – это все же способность модели не реагировать на малые отклонения, а *робастность* – на большие. Поэтому в ряде случаев мы делали такую последовательность действий. В начале решали задачу MILP и отфильтровывали случайные выбросы, а затем на уточненном множестве решали задачу MSE.

Покажем, какие еще дополнительные возможности открываются при сведении задач регрессии к ЗМП. После общения с практиками по задаче настройки агломерационного комплекса были получены комментарии, что некоторые анализируемые параметры физически не могут оказывать разнонаправленное влияние. Например, если агломерационный комплекс состоит из нескольких однотипных агломерационных машин, то коэффициенты регрессии перед одинаковыми по смыслу параметрами (скорость движения ленты, температура за горном, давление в коллекторе и т.п.) должны быть одного знака на всех машинах. Они могут различаться по величине, но не по знаку.

Поэтому модель (1.8–1.9) была дополнена условиями, запрещающими разнонаправленность одинаковых по смыслу параметров:

$$L * (w_i - 1) \leq a_i \leq L * w_i \quad i \in I; \quad (1.10)$$

$$w_g = w_q \quad g \in I, q \in I; \quad (1.11)$$

где g и q – индексы одинаковых по смыслу параметров; w_i – булева переменная.

Тогда, если $w_i = 1$, одинаковые по смыслу переменные больше или равны нулю, а если $w_i = 0$, то одинаковые по смыслу переменные меньше или равны нулю.

Отметим, что введение условий (1.10)–(1.11) не является подгонкой результатов под желаемый сценарий, так как отрицательность или положительность коэффициентов выбирает сама модель исходя из критерия оптимальности. В рассматриваемом случае это минимум суммы непопаданий наблюдений обучающего множества в заданный диапазон.

Конечно, глядя на таблицу, можно сказать, что по MAPE мы незначительно выигрываем у остальных методов и даже проигрываем Random Forest, но результаты Random Forest мы вряд ли смогли бы проинтерпретировать, а в остальные внести условия однонаправленности ряда параметров. Мы ни в коей мере не хотим принизить важность и работоспособность других методов. Просто заранее сказать, какой из методов в конкретной ситуации будет наиболее адекватным, достаточно сложно. Поэтому надо пробовать разные и выбрать наиболее подходящий.

Вообще сведение задач регрессии к ЗМП позволяет шире взглянуть на сами задачи регрессии.

Интересно выяснить, каким количеством уравнений регрессии можно описать обучающую выборку с заданной точностью, т.е. сколько неравенств вида (1.9) требуется, чтобы число наблюдений, не удовлетворяющих ни одному неравенству, было меньше некоторой пороговой величины. Такая постановка может быть описана следующей моделью:

$$\min \sum_{j \in J} \omega_j \quad (1.12)$$

$$(1 - d) * y_j - L * z_j^t \leq \sum_{i \in I} p_{ij} * a_i^t + b^t \leq (1 + d) * y_j + L * z_j^t$$

$$t \in T, j \in J \quad (1.13)$$

$$\sum_{t \in T} z_j^t \leq m - 1 + \omega_j \quad j \in J \quad (1.14)$$

где m — количество уравнений регрессии; t — индекс уравнения регрессии $t = 1 \dots m$; z_j^t — булева переменная (признак непопадания в диапазон t -го уравнения); ω_j — булева переменная (признак непопадания в диапазон всех уравнений).

Отметим, что кусочно-линейная регрессия является частным случаем данной модели. Модель вида (1.12)–(1.14) позволяет выделять практически любые закономерности между входными и выходными параметрами, от описываемых параллельными гиперплоскостями до описываемых гиперплоскостей пересекающимися крест на крест. Такие ситуации не являются чем-то экзотическим, а скорее всего будут свидетельствовать о том, что не учтены какие-то существенные входные параметры.

В кодах CPLEX данная модель выглядит следующим образом:

```

/* НЕСКОЛЬКО УРАВНЕНИЙ РЕГРЕССИИ */
float d=0.1; // допустимый диапазон (доля от выходного параметра)
int m=...; // число линий
range t = 1..m; // индекс линии
int L=10000;
int k=...; // число элементов в множестве
range j = 1..k; // индекс элемента 1-го множества
int n=...; // число переменных (параметры наблюдений)
range i = 1..n; // индекс параметра
float P[j][i]=...; // множество входных параметров объектов
float Y[j]=...; // множество выходных параметров объектов
dvar float a[i][t]; // коэффициенты t-й линии регрессии
dvar float b[t]; // свободный член t-й линии регрессии
dvar boolean z[j][t]; // признак непопадания в диапазон t-й линии регрессии
dvar boolean w[j]; // признак непопадания ни в один из диапазонов
minimize sum(j in j) w[j]; // в ни один диапазон сумма непопаданий
subject to {
forall(t in t) forall(j in j) sum(i in i) P[j][i]*a[i][t]+b[t]<=(1+d)*Y[j]+L*z[j][t];
forall(t in t) forall(j in j) sum(i in i) P[j][i]*a[i][t]+b[t]>=(1-d)*Y[j]-L*z[j][t];
forall(j in j) sum(t in t) z[j][t]<=(m-1)+ w[j]; // нормально, если наблюдение попало хотя бы в один диапазон
};

```

2.3. ЛИНЕЙНО РАЗДЕЛИМЫЕ МНОЖЕСТВА

Имея представление об уравнении прямой и умея решать задачи регрессии, можно переходить к изучению методов классификации. Скажем сразу, что методов классификации достаточно много и основные из них мы рассмотрим. Но для сохранения логики изложения начнем с линейно разделимых множеств. Перед рассмотрением определения данного понятия попробуем для начала рассмотреть конкретную задачу.

Пусть в двухмерном пространстве (то есть на плоскости) заданы два множества MP_1 и MP_2 , содержащие координаты некоторых точек.

Графически множества MP_1 и MP_2 представлены на рис. 5.

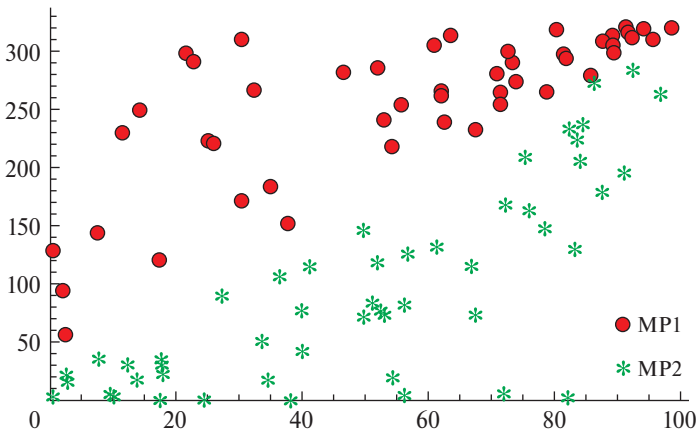


Рис. 5. Пример линейно разделимых множеств

В дальнейшем под линейно разделимыми множествами будем понимать два множества, между которыми в двухмерном пространстве можно провести линию, в трехмерном пространстве плоскость, в n -мерном пространстве гиперплоскость, такую, что точки разных множеств будут расположены по разные стороны линии, плоскости или гиперплоскости.

На рис. 5 видно, что такую линию провести можно. Если это сделать вручную, то получим графическое решение задачи. Попробуем вычислить формулу линии формализованно, а не при помощи линейки и карандаша.

Для удобства математической записи определимся с используемыми символами:

J_1 и J_2 — множества, которые необходимо разделить;

$J = J_1 \cup J_2$ – множество наблюдений;
 I – множество параметров наблюдений;
 i, j – индексы соответствующих множеств;
 p_{ij} – i -й параметр j -го наблюдения (константы);
 a_i – коэффициенты гиперплоскости (искомые переменные);
 b – свободный член гиперплоскости (искомая переменная);
 L – очень большое число;
 ε – очень малое число (используем для строгости ограничений);
 z_j – булевы переменные для фиксации нарушений условий разделения множеств.

В принципе для решения поставленной задачи достаточно найти допустимое решение системы неравенств:

$$\sum_{i \in I} p_{ij} * a_i + b < 0, \quad j \in J_1 \quad (2.1)$$

$$\sum_{i \in I} p_{ij} * a_i + b > 0, \quad j \in J_2, \quad (2.2)$$

где p_{ij} – параметры множеств (числа), a_i – коэффициенты разделяющей линии (в общем случае гиперплоскости), переменные значения, которые надо определить, b – свободный член разделяющей линии, значение которого тоже надо определить. На практике строгие неравенства используются редко, но если их записать как нестрогие в виде

$$\sum_{i \in I} p_{ij} * a_i + b \leq 0, \quad j \in J_1 \quad (2.3)$$

$$\sum_{i \in I} p_{ij} * a_i + b \geq 0, \quad j \in J_2, \quad (2.4)$$

то допустимое решение очевидно (все переменные равны нулю), но совершенно бесполезно. Поэтому неравенства обычно записывают в виде

$$\sum_{i \in I} p_{ij} * a_i + b \leq -\varepsilon, \quad j \in J_1 \quad (2.5)$$

$$\sum_{i \in I} p_{ij} * a_i + b \geq \varepsilon, \quad j \in J_2. \quad (2.6)$$

Возможным также является подход, когда одна система неравенств задается строго, а другая нет, например можно использовать (2.3 и 2.6) или (2.4 и 2.5). Форма записи системы неравенств определяется требованиями, предъявляемыми исследователем к результату, который он желает получить.

В нашем примере $i = 2$, множества J_1, J_2 содержат по 50 элементов. Множества могут быть линейно не разделимы. В этом случае система ограничений будет противоречивой и допустимое решение будет отсутствовать. Поэтому правильнее будет записать систему ограничений в следующем виде:

$$\sum_{i \in I} p_{ij} * a_i + b \leq w_j, \quad j \in J_1 \quad (2.7)$$

$$\sum_{i \in I} p_{ij} * a_i + b \geq -w_j + \varepsilon, \quad j \in J_2 \quad (2.8)$$

$$w_j \geq 0, \quad j \in J, \quad (2.9)$$

где w_j — это возможные корректировки отдельных неравенств.

Вполне естественным является стремление избавиться от таких корректировок. Поэтому добавим целевую функцию (C), например:

$$\min \sum_{j \in J} w_j. \quad (2.10)$$

Если множества линейно разделимы, то $C = 0$, иначе $C > 0$.

Если $C > 0$, то исследователь может продолжить работу с корректировками. Например, потребовать, чтобы максимальная из них была минимальной (критерий минимакса). Для этого необходимо ввести условия: $w_j \leq u$, $j \in J$, а целевую функцию записать в виде: $\min u$.

Так как в данном примере нас интересуют не величины w_j , а сам факт, выполняется условие разделения множеств или нет, то правильным будет ввести следующие условия:

$$w_j \leq L * z_j \quad j \in J, \quad (2.11)$$

где z_j — булева переменная, принимающая значение 0, если условие разделения множеств выполняется, и 1 в противоположном случае. Таким образом, переменная z_j будет фиксировать факт нарушения условия разделения. Соответственно целевая функция будет

$$\min \sum_{j \in J} z_j. \quad (2.12)$$

В более компактном виде модель может быть представлена как

$$\sum_{i \in I} p_{ij} * a_i + b - L * z_j \leq 0, \quad j \in J_1 \quad (2.13)$$

$$\sum_{i \in I} p_{ij} * a_i + b + L * z_j \geq \varepsilon, \quad j \in J_2 \quad (2.14)$$

$$\min \sum_{j \in J} z_j. \quad (2.15)$$

Попробуем дать не геометрическую, а другую, содержательную интерпретацию. Представим, что существует некий уникум, способный видеть, как расположены точки в пространстве любой размерности. Тогда его мнения будет достаточно, к какому из множеств отнести любую точку. Данный случай можно трактовать как существование комитета, состоящего из одного члена, единолично принимающего решения. Параллели в обществе — диктатор, единоличный никому не доверяющий собственник, и тому подобные. Их роль в данном случае выполняет гиперплоскость.

Отметим также, что если исследуется только вопрос линейной делимости множеств, то для одного из множеств система неравенств может быть задана без корректировок. В дальнейшем мы будем поступать таким образом при построении комитета старшинства итерационным способом.

2.4. МЕТОД ОПОРНЫХ ВЕКТОРОВ И ПОТЕНЦИАЛЬНЫХ ФУНКЦИЙ

Выше мы уже отмечали, что и в случае линейной делимости множеств не все так просто, так как если множества линейно делимы, то в общем случае число гиперплоскостей, их разделяющих, равно бесконечности и возникает закономерный вопрос, а какое решение лучше. Вполне логично будет считать в качестве лучшей гиперплоскости, разделяющей множества, равноудаленную от каждого множества.

Алгоритм построения такой гиперплоскости был предложен В. Вапником и А. Червоненкисом [5] в 1963 г. По сути они свели данную задачу к задаче **квадратичного программирования** и решили ее на основе теоремы Куна—Таккера. Впоследствии, в 1992 г. В. Вапник, Б. Босер и И. Гийон предложили способ создания нелинейного классификатора на основе перехода от скалярных произведений к произвольным ядрам, т.е. разделение линейным классификатором в другом пространстве признаков. Описанный выше подход называется Метод опорных векторов (англ. Support Vector Machine, SVM).

Перейдем к рассмотрению метода потенциальных функций (МПФ). Часто пишут, что МПФ является частным случаем метода ближайших соседей. Наверное, более правильно будет сказать, что данный метод является дальнейшим развитием метода ближайших соседей. Идея метода пришла скорее всего из физики. Для понимания данного подхода вспомним ряд физических законов. Начнем с законов Ньютона, открытых еще в 1666 г., — с закона всемирного тяготения и принципа суперпозиции, которые звучат следующим образом:

Сила гравитационного притяжения между двумя материальными объектами с массами m_1 и m_2 , разделенными расстоянием r , действует вдоль соединяющей их прямой, пропорциональна обоим массам и обратно пропорциональна квадрату расстояния между ними:

$$F = \gamma * \frac{m_1 * m_2}{r^2},$$

где γ – гравитационная постоянная, т.е. константа.

Принцип суперпозиции: Результат воздействия на объект нескольких внешних сил есть векторная сумма этих сил.

Вспомним закон Кулона из области электростатики, открытый им в 1785 г.:

Модуль силы взаимодействия двух точечных зарядов в вакууме прямо пропорционален произведению модулей этих зарядов и обратно пропорционален квадрату расстояния между ними:

$$F = k * \frac{q_1 * q_2}{r^2},$$

где k – коэффициент пропорциональности, т.е. константа.

Внешне закон всемирного тяготения и закон Кулона совпадают один в один. Мы не будем далее продолжать перечисление других физических законов, но отметим, что во многих из них сила взаимодействия объектов друг с другом убывает пропорционально квадрату расстояния между ними.

В наиболее общем виде функция называется потенциальной, если она резко убывает с ростом аргумента. Например, $F = \frac{1}{r^2}$, где F – функция, r – аргумент функции. Данную функцию неудобно использовать, так как при $r = 0$ она стремится к бесконечности. Поэтому на практике обычно используют функцию: $F = \frac{1}{1 + r^2}$.

Тогда, если $r = 0$, то $F = 1$, а с ростом r F резко убывает. Если значения r большие и с очень маленькими F работать неудобно, то в числителе вместо 1 можно использовать некоторую положительную константу.

Основная идея метода потенциальных функций (МПФ) состоит в следующем. Считаем, что каждая точка любого множества генерирует некоторый потенциал на все остальные точки, как в законе всемирного тяготения или Кулона. Суммарный потенциал в каждой точке определяется как сумма потенциалов, исходящих от всех точек (принцип суперпозиции). Необходимо подобрать весовые коэффициенты для каждой точки таким образом, чтобы суммарные потенциалы для точек одного множества были положительными, а для другого отрицательными.

Обычно в научных статьях для решения задач МПФ предлагаются различного рода итерационные алгоритмы, и сами авторы отмечают, что эти алгоритмы просты в реализации, но имеют плохую сходимость. Потому мы предлагаем представить данную задачу как ЗМП и решить ее стандартными методами. Более того, для решения задач относительно небольшой размерности нам даже не понадобятся пакеты математического программирования, так как задача может быть представлена как задача линейной алгебры, для решения которой требуется только вычислить обратную матрицу. Последовательность действий при таком подходе следующая:

1. Вычисляем квадраты евклидовых расстояний между всеми точками обучающей выборки.
2. На основе квадратов евклидовых расстояний вычисляем потенциалы.
3. Вычисляем обратную матрицу или решаем задачу линейного программирования для определения весовых коэффициентов всех точек.

Дадим ряд пояснений. Очевидно, что матрица потенциалов квадратная. **Поэтому система уравнений:** $F^*a = b$, где F – матрица потенциалов, a – вектор весовых коэффициентов наблюдений, b – некоторый произвольный ненулевой вектор; **всегда имеет решение, если существует матрица F^{-1} , т.е. обратная к F .** Поэтому, если в векторе b элементам одного множества будет присвоено значение -1 , а другого 1 , то система будет иметь решение, если существует F^{-1} .

Давайте порассуждаем. Если мы используем k -NN, то считать ли каждый объект соседом самому себе? Очевидно, что нет. Тогда, используя МПФ, надо ли учитывать потенциал, который объект генерирует сам на себя? Естественно, тоже нет. Поэтому надо просто обнулить потенциалы на диагонали матрицы потенциалов и можно делать расчеты.

Конечно, данную задачу лучше решать как ЗМП. Поэтому приведем ее запись сразу в кодах IBM ILOG CPLEX:

```
/*ДЛЯ КАЖДОЙ ТОЧКИ ОБУЧАЮЩЕЙ ВЫБОРКИ НЕОБХОДИМО ПОДОБРАТЬ ВЕС ТАКИМ ОБРАЗОМ, ЧТОБЫ СУММАРНЫЙ ПОТЕНЦИАЛ ДЛЯ КАЖДОЙ ТОЧКИ ОДНОГО МНОЖЕСТВА БЫЛ ПОЛОЖИТЕЛЬНЫМ, А ДЛЯ КАЖДОЙ ТОЧКИ ДРУГОГО ОТРИЦАТЕЛЬНЫМ */
```

```
int k = ...; // число наблюдений (объектов) в множестве1  
range i = 1..k; // индекс наблюдения (объекта)  
int m = ...; // число наблюдений (объектов) в множестве2  
range j = 1..m; // индекс наблюдения (объекта)  
int n = ...; // суммарное число наблюдений (объектов)  
range s = 1..n; // индекс наблюдения (объекта)
```

```

float F1[i][s]=...; // потенциалы для точек множества 1 ( $C/(1+R^2[i][s])$ ), // где C некоторая положительная константа,
R-евклидово расстояние
float F2[j][s]=...; // потенциалы для точек множества 2

```

```

/* искомые переменные */
dvar float a[s]; // ВЕС ТОЧКИ В СУММАРНОМ ПОТЕНЦИАЛЕ
dvar float+ v[i]; // невязка
dvar float+ w[j]; // невязка

minimize sum(i in i) v[i]+sum(j in j) w[j];
subject to {
forall(i in i) sum(s in s) F1[i][s]*a[s]<= -1+v[i];
forall(j in j) sum(s in s) F2[j][s]*a[s]>= 1-w[j];
};

```

Вообще, строго говоря, в методе ближайших соседей нет никакого обучения, так как надо хранить всю обучающую выборку и каждый раз считать расстояние от тестируемой точки до всех точек обучающей выборки. Кроме того, всегда возникает вопрос: на сколько ближайших соседей надо ориентироваться? Это и есть его основные недостатки, но k-NN достаточно надежен и хорошо интерпретируем. Поэтому им часто пользуются практические работники.

В отличие от k-NN МПФ позволяет исключать неинформативные наблюдения. Поэтому он более экономен по объему хранимой информации и времени на разметку контрольной выборки.

2.5. ОЦЕНКА КАЧЕСТВА РЕШЕНИЯ В ЗАДАЧАХ КЛАССИФИКАЦИИ

Допустим, мы решили задачу классификации несколькими методами и нам надо выбрать наилучший. Для того чтобы это можно было сделать, необходимы формализованные оценки качества решения (метрики) [57–65]. Хотя основой проверки качества решающего правила является тестовая выборка, способы оценки качества решения применимы как к тестовой, так и к обучающей выборкам. Рассмотрим возможные исходы классификации конкретного объекта:

<i>Результат классификации</i>		<i>Сокращение</i>
Истинно-положительный	True-positive	TP
Ложно-положительный	False-positive	FP
Истинно-отрицательный	True-negative	TN
Ложно-отрицательный	False-negative	FN

В статистике FP принято называть ошибкой первого рода, а FN – ошибкой второго рода.

Приведем примеры для двух случаев. В первом алгоритм классификации определяет здоровый/больной, во втором спам/неспам. Сравнение результатов классификатора и реальных данных приведено в таблице.

	Был классифицирован как	В действительности оказался
TP	1. Больной 2. Спам	1. Больной 2. Спам
FP	1. Больной 2. Спам	1. Здоровый 2. Неспам
TN	1. Здоровый 2. Неспам	1. Здоровый 2. Неспам
FN	1. Здоровый 2. Неспам	1. Больной 2. Спам

Пусть конкретная выборка состоит из P объектов класса Positive и N объектов класса Negative. Какой класс называть положительным или отрицательным, зависит от конкретной задачи и сложившихся традиций. Если вам скажут, что ваш анализ на COVID-19 дал положительный результат, не спешите радоваться. Это означает, что скорее всего Вы больны, но не стоит и паниковать, так как диагноз может быть ложно-положительным, все зависит от точности теста.

Далее, под TP, FP, FN, TN будем понимать сумму каждого исхода по выборке, для которой оцениваем качество решающего правила.

Таблицу результатов классификатора удобно представлять в следующем виде:

	Оценка классификатора Positive	Оценка классификатора Negative	Итого реальных объектов в классе
Реальный класс Positive	TP	FN	TP+FN
Реальный класс Negative	FP	TN	FP+TN
Итого объектов согласно классификатору	TP+FP	FN+TN	

По сути дела все метрики строятся на основе данной таблицы, которую принято называть матрица ошибок. Первое, что чисто интуитивно приходит в голову, это определить долю правильных классификаций в общем количестве, т.е. определить вероятность того, что классы будут предсказаны **правильно**. Данную метрику по-русски так и будем называть **правильность (accuracy)**, потому что если ее перевести как **точность**, то это приведет к путанице в названиях, тогда уж лучше переводить **аккуратность**.

$$Accuracy = \frac{\text{сумма правильных классификаций}}{\text{общее число классифицируемых объектов}} =$$

$$= \frac{TP + TN}{TP + TN + FP + FN}.$$

Данная метрика хорошо воспринимается практическими специалистами и достаточно хорошо работает, если классы сбалансированы, но в случае значительной несбалансированности классов данная метрика может стать совершенно бесполезной, так как можно просто всегда предсказывать только класс со значительно большим количеством объектов и иметь хороший показатель ассурасу. Поэтому на практике используются другие метрики. Разобьем их на две большие группы.

Показатели, вычисляемые только на основе матрицы ошибок.

Варианты названия	Формулы
True Positive Rate, Recall, Sensitivity, Probability of detection Полнота, чувствительность, вероятность выявления	$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$
False Positive Rate	$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}$
True Negative Rate, Specificity Специфичность	$TNR = \frac{TN}{N} = \frac{TN}{TN + FP}$
False Negative Rate Вероятность ложной тревоги	$FNR = \frac{FP}{N} = \frac{FP}{TN + FP} = 1 - TNR$
Precision Точность	$\frac{TP}{TP + FP}$
Jaccard index	$\frac{TP}{TP + FN + FP}$
Accuracy, правильность	$\frac{TP + TN}{P + N}$

Принцип составления данных метрик очень прост. В числителе один или сумма нескольких секторов матрицы ошибок, в знаменателе — то же, что и в числителе, плюс некоторая добавка в виде сектора или секторов, которые не вошли в числитель. Поэтому значение всех этих метрик от нуля до единицы. Если вы забыли название конкретной метрики, то говорите «доля» и называете то, что стоит в числителе, «в», и называете то, что стоит в знаменателе. Так обычно

и делают, когда хотят пояснить суть метрики. Вы можете сами сгенерировать новые метрики, например $\frac{FP}{FP + TP}$, но это $1 - Precision$, поэтому в ней нет необходимости. Некоторая ирония, присутствующая в данном абзаце, связана с тем, что в разных статьях одни и те же метрики могут называться по-разному и это может затруднять восприятие. Поэтому смотрите на формулу, а не на название.

Очевидно, что самым наилучшим классификатором будет тот, который разделит классы без ошибок. В этом случае $FP = 0$ и $FN = 0$, а точность, полнота, специфичность, ассигасу будут равны единице. На практике такое бывает редко, поэтому все перечисленные показатели обычно меньше единицы.

Так как оценка качества решающих правил важна для выбора, каким из них пользоваться в дальнейшем, существуют и другие показатели, которые строятся на основе приведенных выше.

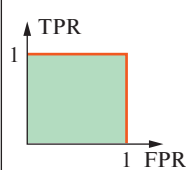
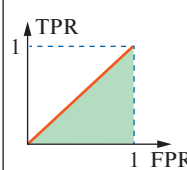
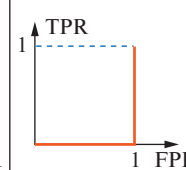
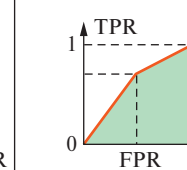
Взвешенное гармоническое среднее	$F_{\beta} = (1 + \beta^2) * \frac{\text{Precision} * \text{recall}}{\beta^2 * \text{Precision} + \text{recall}}$, если β – любое
Гармоническое среднее	$F = 2 * \frac{\text{Precision} * \text{recall}}{\text{Precision} + \text{recall}}$, если $\beta = 1$
Geometric mean (GM)	$\sqrt{\text{precision} * \text{recall}}$
Каппа of Cohen Каппа Коэна	$\frac{(TP + FP)(TP + FN) + (FN + TN)(FP + TN)}{(P + N)^2}$
Index of Youden Индекс Юдена	Sensitivity+ Specificity-1
Diagnostic odds ratio(DOR) Диагностическое отношение шансов	$\frac{TP}{FN} / \frac{FP}{TN} = \frac{\text{Sensitivity}}{1 - \text{Sensitivity}} / \frac{1 - \text{Specificity}}{\text{Specificity}}$
Discriminatory power (DP) Степень разделимости	$k \left(\log \frac{\text{Sensitivity}}{1 - \text{Sensitivity}} + \log \frac{\text{Specificity}}{1 - \text{Specificity}} \right)$
Area Under ROC Curve (AUC) Площадь под ROC-кривой	$\frac{\text{Sensitivity} + \text{Specificity}}{2}$

Интересную статистику по использованию различных метрик приводит П.В. Дудченко [13]. Им рассматривались научные публикации на английском языке по применению методов машинного обучения в медицине за период с 2011 по 2018 г. Результаты приведены в следующей таблице:

Метрика	Количество публикаций
AucROC	29
Полнота, чувствительность	25
Специфичность	22
Ассурасу	18
F-мера	6
Точность	5

Из таблицы видно, что наиболее часто используемой метрикой является площадь под ROC-кривой (AucRoc). Это связано с тем, что данная метрика наименее чувствительна к дисбалансу классов. Поясним более подробно, что такое ROC-кривая и зачем надо вычислять площадь под кривой. ROC-кривые в годы Второй мировой войны использовались для повышения качества распознавания самолетов противника по радиолокационному сигналу и служили для отображения соотношения между долей обнаруженных с помощью радара целей и долей ложной тревоги. Receiver Operating Characteristic переводится как операционная характеристика приемника.

Так как максимальные значения FPR и TPR равны единице, а минимальные равны нулю, представим квадрат размером 1×1 с вершиной в начале координат. Для случая **бинарных ответов алгоритма**, под ROC-кривой будем понимать линию в координатах $x = FPR$ и $y = TPR$, последовательно соединяющую точки $(0,0)$; (FPR,TPR) и $(1,1)$. Четыре возможных ситуации приведены в следующей таблице. Под AUC будем понимать площадь квадрата, лежащую ниже красной линии.

Идеальный	Монетка	Наихудший	Общий
FPR = 0 TPR = 1 AUC = 1	FPR = 0,5 TPR = 0,5 AUC = 0,5	FPR = 1 TPR = 0 AUC = 0	$AUC = \frac{1 + TPR - FPR}{2}$
			

В случае идеального (безошибочного) классификатора $TPR = 1$, $FPR = 0$ площадь по кривой (AUC) равна 1. В случае наихудшего $AUC = 0$. Для общего случая

$$\begin{aligned} AUC &= \frac{FTR * FPR}{2} + TPR * (1 - FPR) + \frac{(1 - TPR) * (1 - FPR)}{2} = \\ &= \frac{TPR + 1 - FPR}{2} = \frac{\text{Sensitivity} + \text{Specificity}}{2}. \end{aligned}$$

AUC можно интерпретировать как вероятность того, что классификатор правильно разделит классы. Если $AUC = 0,5$, то он ничем не лучше случайного угадывания. Поэтому в таблице данный вариант называется Монетка. Если $AUC < 0,5$, то классификатор работает хуже случайного. Таким образом, для дальнейшего использования имеет смысл рассматривать только классификаторы с $AUC > 0,5$, причем чем больше AUC, тем качественнее классификатор.

Рассмотрим возможность максимизации AUC сразу при построении классификатора. Так как теоретически возможный максимум равен единице, немного преобразуем критерий и будем искать

$$\min\left(1 - \frac{1 + TPR - FPR}{2}\right) = \frac{1 - TPR + FRP}{2}.$$

Естественно, нам необходимо минимизировать числитель. Распишем его подробнее:

$$\begin{aligned} \min\left(1 - \frac{TP}{TP + FN} + \frac{FP}{FP + TN}\right) &= \left(1 - \frac{TP}{P} + \frac{FP}{N}\right) = \\ &= \left(\frac{P * N - N * TP + P * FP}{P * N}\right). \end{aligned}$$

Допустим, мы пытаемся разделить классы гиперплоскостью или нелинейной гиперповерхностью. Так как многие задачи разделения нелинейной гиперповерхностью сводятся к разделению гиперплоскостью в другом пространстве признаков, мы больше не будем отдельно говорить про этот случай. Итак, разделяем гиперплоскостью. Задача сводится к задаче линейного программирования с частично булевыми переменными. Сформулируем систему ограничений:

$$\sum_{i \in I} p_{ij} * a_i + b \leq -1 + L * z_j, \quad j \in J_1, \quad (2.16)$$

$$\sum_{i \in I} p_{ij} * a_i + b \geq 1 - L * z_j, \quad j \in J_2, \quad (2.17)$$

где J_1 – множество *Positive*; J_2 – множество *Negative*; I – множество признаков; i, j – индексы соответствующих множеств p_{ij} – i -й признак j -го объекта; b – свободный член; L – большое число; z_j – булева переменная, равная 1, если j -й объект относится к соответствующему классу и 0 если и не относится.

Тогда матрица ошибок будет выглядеть следующим образом:

$TP = P - \sum_{j \in J_1} z_j$	$FN = \sum_{j \in J_1} z_j$
$FP = \sum_{j \in J_2} z_j$	$TN = N - \sum_{j \in J_2} z_j$

Целевую функцию мы можем записать так:

$$\begin{aligned} \min P * N - N * (P - \sum_{j \in J_1} z_j) + P * \sum_{j \in J_2} z_j = \\ = N * \sum_{j \in J_1} z_j + P * \sum_{j \in J_2} z_j. \end{aligned} \quad (2.18)$$

Таким образом, задача максимизации AUC сводится к ЗМП вида (2.16)–(2.18). Если в результате решения мы получили AUC, которое нас не устраивает, можем улучшать решение за счет увеличения количества гиперплоскостей и построения комитетных конструкций.

В случае если ответы алгоритма не бинарные, а вероятностные, для вычисления AUC необходимо учесть разметку классификатором всех объектов выборки. Построение графика будет сложнее, но суть останется та же. В наиболее общем виде $AUC = \int_0^1 TPR dFPR$.

КОМИТЕТНЫЕ ПОДХОДЫ К РЕШЕНИЮ ЗАДАЧ КЛАССИФИКАЦИИ

3.1. ЛИНЕЙНО НЕРАЗДЕЛИМЫЕ МНОЖЕСТВА И МЕТОД КОМИТЕТОВ

Как правило, в реальной жизни приходится работать в условиях линейно неразделимых множеств, когда в n -мерном пространстве между множествами нельзя провести гиперплоскость.

Графически это показано на рис. 6.

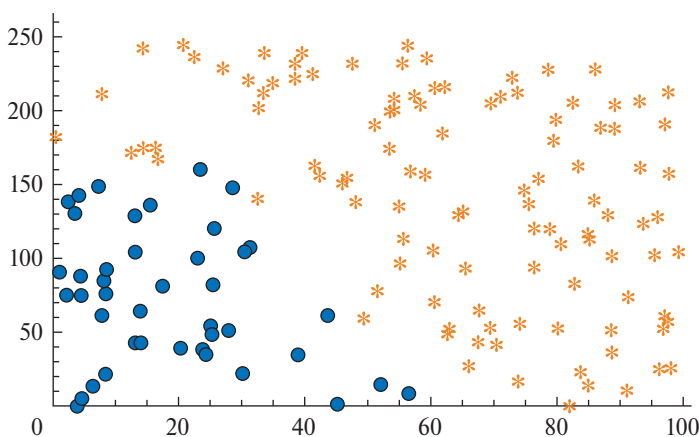


Рис. 6. Пример линейно неразделимых множеств

Пока не будем рассматривать возможность разделения множеств нелинейными гиперповерхностями, так как такие решения могут быть излишне сложными. Естественно, если нельзя разделить одной, то может быть можно разделить несколькими. Такой метод называется методом комитетов.

Метод комитетов относится к методам классификации с учителем, применяемым для решения проблемы отнесения того или

иногo объекта к одному из двух классов объектов. Наличие учителя означает, что предварительно существует некоторая обучающая выборка, на которой нам известно, какой объект к какому классу относится. Сама логика работы метода напоминает логику работы «обычного комитета» как коллегиального руководящего органа, где решение принимается на основании решений группы экспертов. В случае метода комитетов роль экспертов исполняют несколько разделяющих гиперплоскостей, называемых членами комитета, каждая из которых голосует за решение по-своему. Итоговое решение принимает комитет в целом, принимая во внимание решение всех отдельных членов за счет использования логики комитетов.

Для лучшего понимания указанного определения вернемся к рис. 6. Как уже было сказано, первой линией разграничить множества действительно нельзя. Если бы мы использовали алгоритм линейного разделения, то нами был бы получен результат с тремя ошибками, как показано на рис. 7.

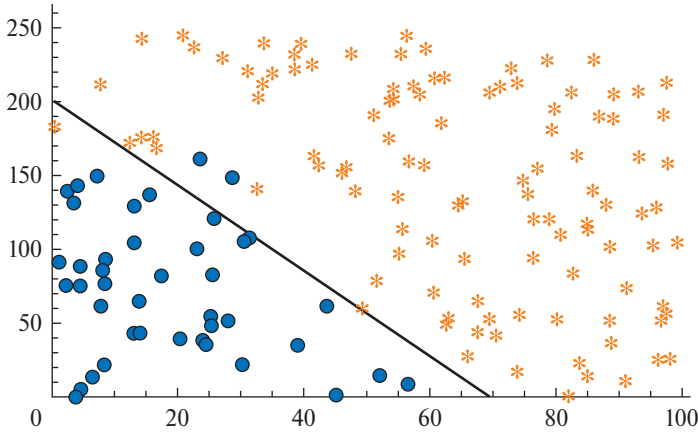


Рис. 7. Линейное разделение линейно неразделимых множеств

Однако возможно разделение и без ошибок. Для этого только и нужно, что воспользоваться двумя линейными классификаторами. Для начала сделаем такое разграничение с помощью карандаша и линейки (рис. 8). Обратим внимание на то, что все точки одного из множеств лежат ниже каждой из линий.

Если опять провести параллели с обществом, то данный случай можно интерпретировать как наличие комитета из двух членов. Очевидно, что если существует несколько членов комитета, то решение будет однозначным только в случае, если четко заранее прописана процедура голосования. Опять же, на примере общества мы

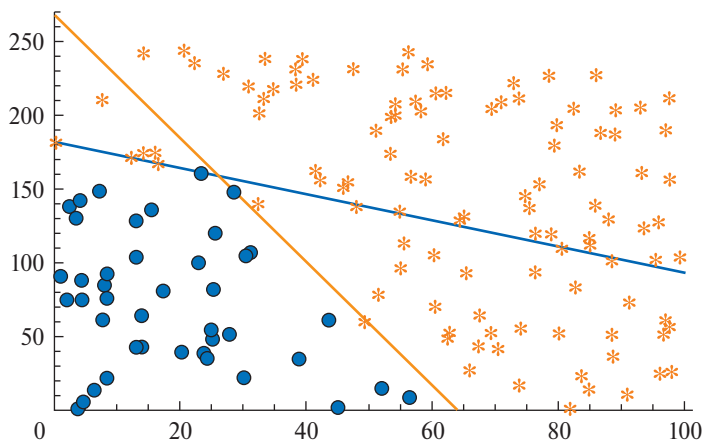


Рис. 8. Разделение линейно неразделимых множеств композицией линейных разделителей

знаем, что голоса их могут быть равнозначными и неравнозначными, т.е. существуют старший и младший члены комитета.

Если их голоса равнозначны, то решение может быть принято, например, в случае их единогласного голосования за или против, поэтому данный комитет принято называть комитетом единогласия (далее по тексту КЕ). Геометрически это означает, что точка лежит ниже каждой из линий (гиперплоскостей).

В реальной жизни примером такого комитета может служить принцип парламентского устройства в Речи Посполитой, который позволял любому депутату сейма прекратить обсуждение вопроса в сейме и работу сейма вообще, выступив против. Данный принцип действовал более 200 лет. Из современных примеров можно привести процедуру принятия решения по непроцедурным вопросам в Совете Безопасности ООН. Решение здесь требует также единогласия со стороны постоянных членов Совета Безопасности.

Достаточно часто при равнозначности голосов разделить множества при помощи КЕ не представляется возможным. В таком случае вместо единогласного решения можно пытаться найти «коллектив наблюдений» такой, что за удовлетворение каждого ограничения будут «голосовать» большинство членов комитета. Такой комитет будет называться комитетом большинства (КБ). Самый известный пример КБ из реальной жизни – это выборы президента РФ. Если рассматривать выборы в терминологии метода комитетов, то данная задача представляет собой классификацию множества кандидатов в президенты на множества «президент» и «непрезидент». Соответственно в данной терминологии каждого избирателя можно

представить, как отдельный член комитета, который на основании своего жизненного опыта и известных ему параметров кандидатов голосует за того или иного кандидата.

В случае, когда голоса неравнозначны, процедура может быть следующей. Если старший член проголосовал «за», то решение принято. Если старший член воздержался, то он делегировал принятие решения младшему члену. Аналог ситуации, когда шеф, не желая вникать в мелкий для него вопрос, говорит подчиненному: «Примите решение самостоятельно». Такой комитет называется комитетом старшинства (далее по тексту КС), и именно с него мы начнем рассмотрение комитетных конструкций. Такой выбор сделан по очень простой причине – на основе навыков по линейному разделению множеств читатель легко может построить итерационные алгоритмы построения КС, а мы ему в этом поможем.

3.2. ИТЕРАЦИОННЫЙ АЛГОРИТМ РАЗДЕЛЕНИЯ НА ОСНОВЕ КОМИТЕТА СТАРШИНСТВА

Если внимательно посмотреть на предыдущий рисунок, то естественным образом возникает желание, используя гиперплоскости, отсечь элементы одного множества от другого. То есть мы требуем, чтобы для первого множества условия выполнялись строго и жестко (без корректировок). Процедуру повторяем два раза. Проиллюстрируем это графически на примере множеств из рис. 6.

На первой итерации проводим линию $y_1 = 290,95 - 4,9863x$ (рис. 9).

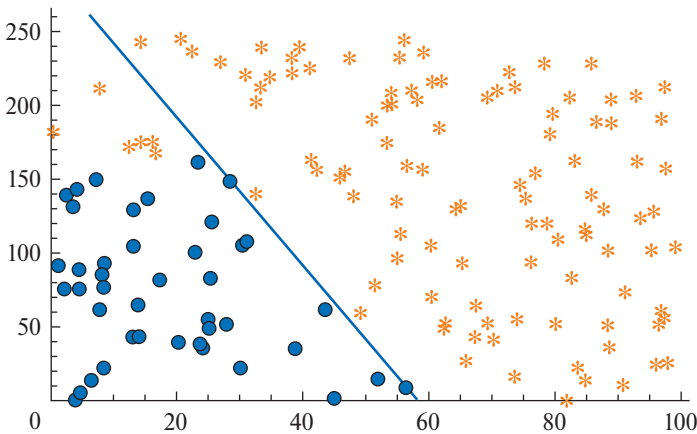


Рис. 9. Проведение разделяющей линии y_1

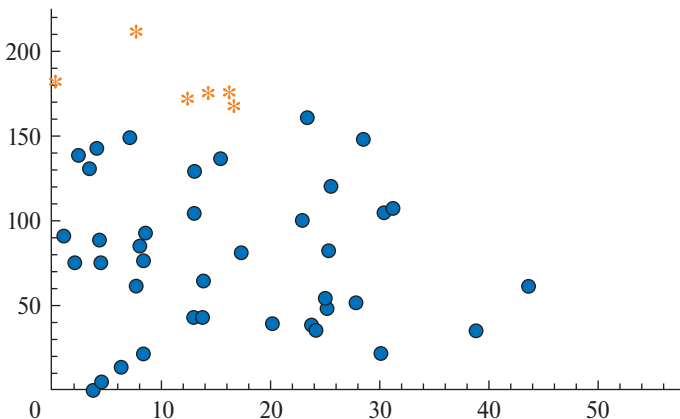


Рис. 10. Множества после отсечения на первой итерации

Как видно на рисунке, количество нарушений условий *равно шести*.

На рис. 10 отсечем полученное на первой итерации множество.

На второй итерации проведем линию $y_2 = 183,91 - 0,96352x$ (рис. 11).

Из рис. 11 видно, что количество нарушений условий *равно нулю*. Значит, множества линейно разделимы, соответственно завершаем итерационный процесс. Общее отображение данной задачи представлено на рис. 12.

Формулировка КС в данном случае будет следующая:

1. Если точка расположена выше линии y_1 , то она относится к множеству МР2, в противном случае решение принимает младший член.

2. Если точка расположена выше линии y_2 , то она относится к множеству МР2, в противном случае – к МР1.

Наблюдательный читатель, думаем, заметил, что на основе этих двух линий можно сформулировать и КЕ в следующем образе: если точка расположена одновременно ниже y_1 и y_2 , то она относится к множеству МР1, в противном случае – к МР2. Естественно, что такие совпадения не обязательны.

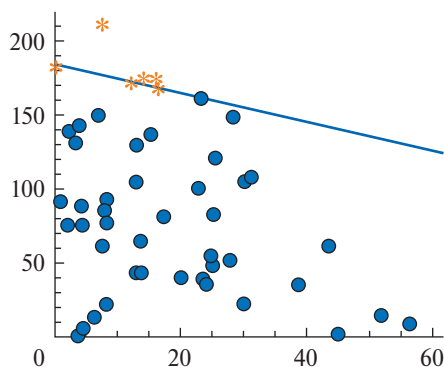


Рис. 11. Проведение разделяющей линии y_2

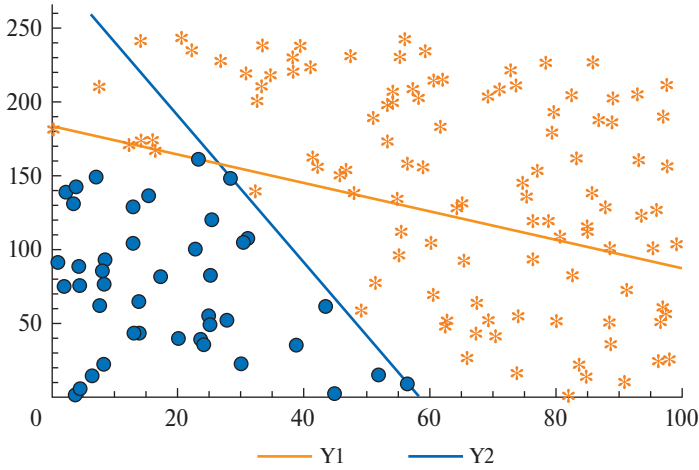


Рис. 12. Разделение множеств двумя линейными классификаторами

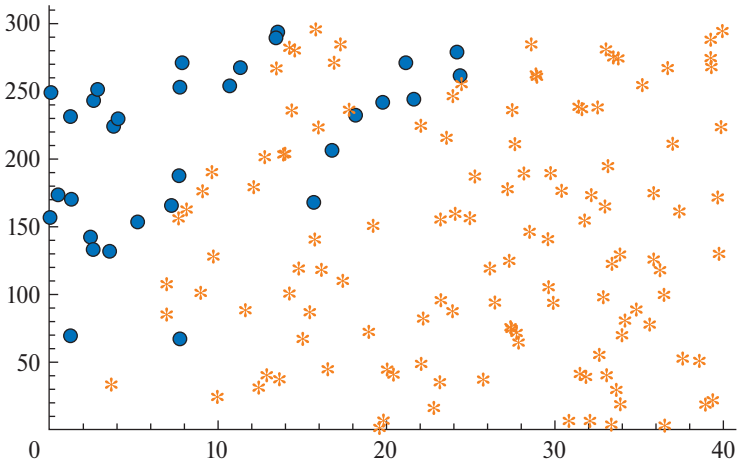


Рис. 13. Пример множеств, разделяемых КС

Рассмотрим более сложные примеры использования итерационных алгоритмов для построения КС. Последовательность шагов проиллюстрируем графически. Первоначальные множества соответствуют изображению на рис. 13.

На первой итерации проводим линию $y_1 = 11,096 + 19,118x$ (рис. 14).

На рис. 15 отсечем полученное на первой итерации множество.

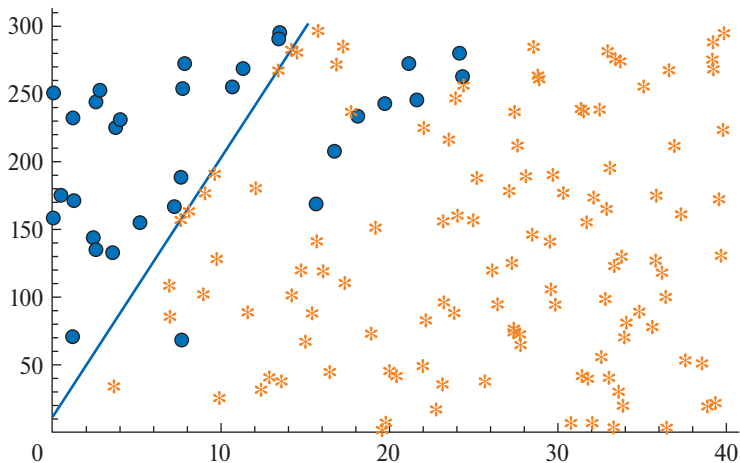


Рис. 14. Проведение разделяющей линии u_1

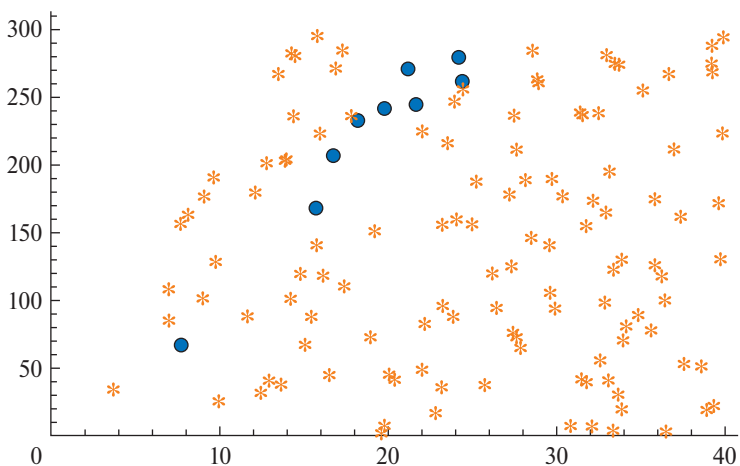


Рис. 15. Множества после отсечения на первой итерации

На второй итерации разделяющая линия $y_2 = -22,109 + 11,65x$ (рис. 16).

На рис. 17 отсечем полученное на второй итерации множество.

На третьей итерации разделяющая линия $y_3 = -24,344 + 14,161x$ (рис. 18). Из рис. 18 видно, что множества линейно разделимы. Соответственно завершаем итерационный процесс. Общее отображение задачи представлено на рис. 19.

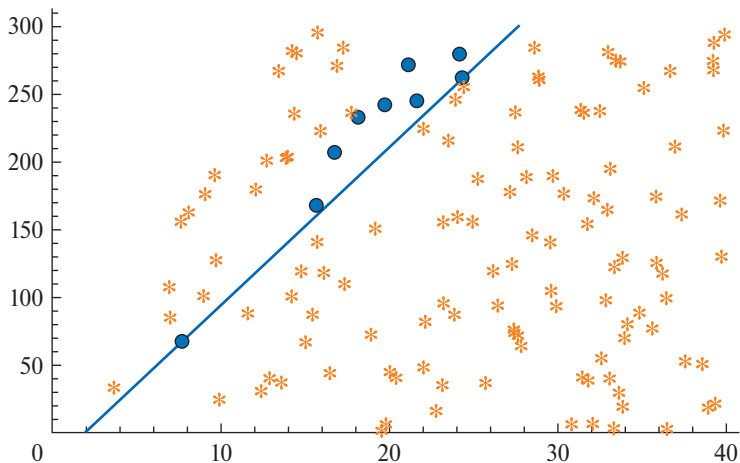


Рис. 16. Проведение разделяющей линии y_2

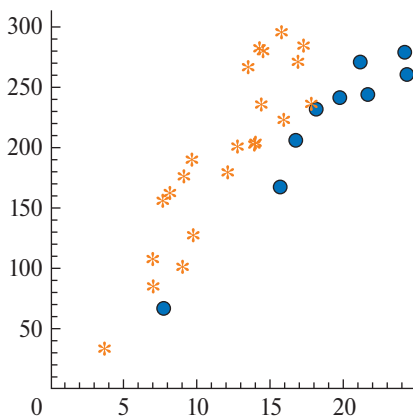


Рис. 17. Множества после отсечения на второй итерации

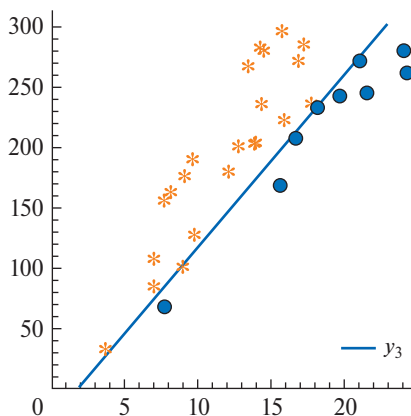


Рис. 18. Проведение разделяющей линии y_3

На всех итерациях использовалась та же модель, что и ранее, только на второй итерации строгие и жесткие ограничения задавались для множества MP_2 .

Формулировка комитета старшинства в данном случае будет следующая:

1) Если точка расположена выше линии y_1 , то она относится к множеству MP_1 , в противном случае решение принимает следующий по рангу член.

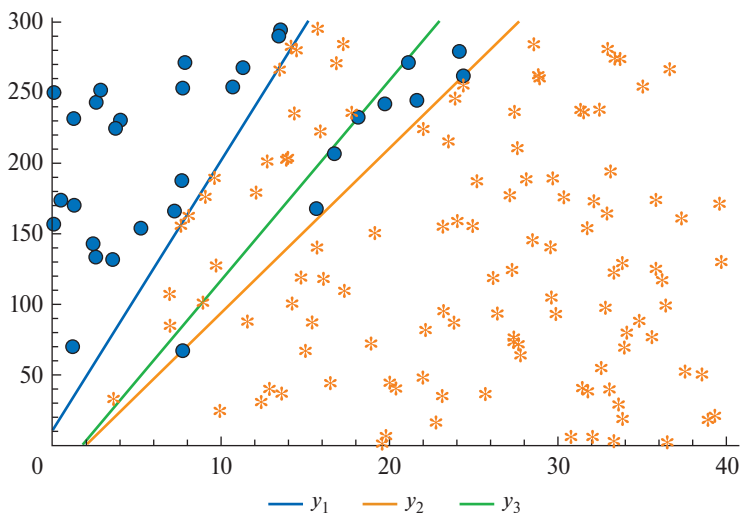


Рис. 19. Разделение множеств тремя линейными классификаторами

2) Если точка расположена ниже линии y_2 , то она относится к множеству МР2, в противном случае решение принимает следующий по рангу член.

3) Если точка расположена ниже линии y_3 , то она относится к множеству МР1, в противном случае она относится к множеству МР2.

Если опять провести параллели с обществом, то при наличии трех членов комитета, даже в случае равнозначности голосов, голосование можно организовать по принципу большинства. Несложно заметить, что на основе этих трех линий можно сформулировать КБ. Зададим каждой линии направление голосования относительно множества МР1. Для y_1 и y_3 это будет направление вверх, для y_2 – вниз. Если пространство разбить на четыре сектора (римские цифры сверху), то для каждого сектора итоги голосования будут соответствовать результатам, представленным в табл. 1.

Таблица 1

Итоги голосования по секторам

Сектор	За	Против
I	y_1, y_2	y_3
II	y_2	y_1, y_3
III	y_2, y_3	y_1
IV	y_3	y_1, y_2

Из таблицы видно, что каждая точка однозначно идентифицируется по принципу большинства. Естественно, что совпадение членов комитета старшинства и комитета большинства тоже не обязательно.

В работах В.Д. Мазурова доказано, что КБ существует всегда, но может состоять из большого числа членов. Рассмотрим пример, когда КС нельзя свести к КЕ и не очевидно, как свести его к КБ. Графически это можно проиллюстрировать следующим образом. На рис. 20 приведены множества МР1 (63 элемента) и МР2 (88 элементов).

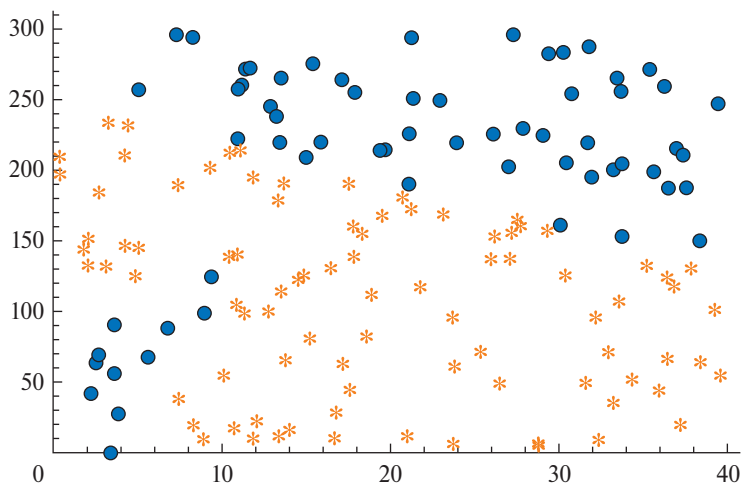


Рис. 20. Пример множеств КС

На первой итерации сперва выбираем разделяющую линию u_1 , для этого для сравнения проведем следующие два уравнения прямой (рис. 21).

Из рис. 21, а можно видеть, что для первого варианта количество нарушений условий равно 55. Соответственно, если воспользоваться данным решающим правилом, то останется классифицировать 118 элементов ($55 + 63$). При этом для второго варианта (рис. 21, б) количество нарушений условий равно 11. Если воспользоваться данным решающим правилом, то останется классифицировать 99 элементов ($11 + 88$).

Выбираем множество с наименьшим количеством оставшихся элементов.

Количество оставшихся элементов множества МР1 равно 11. На рис. 22 отсечем полученное на первой итерации множество, которое не содержит нарушений.

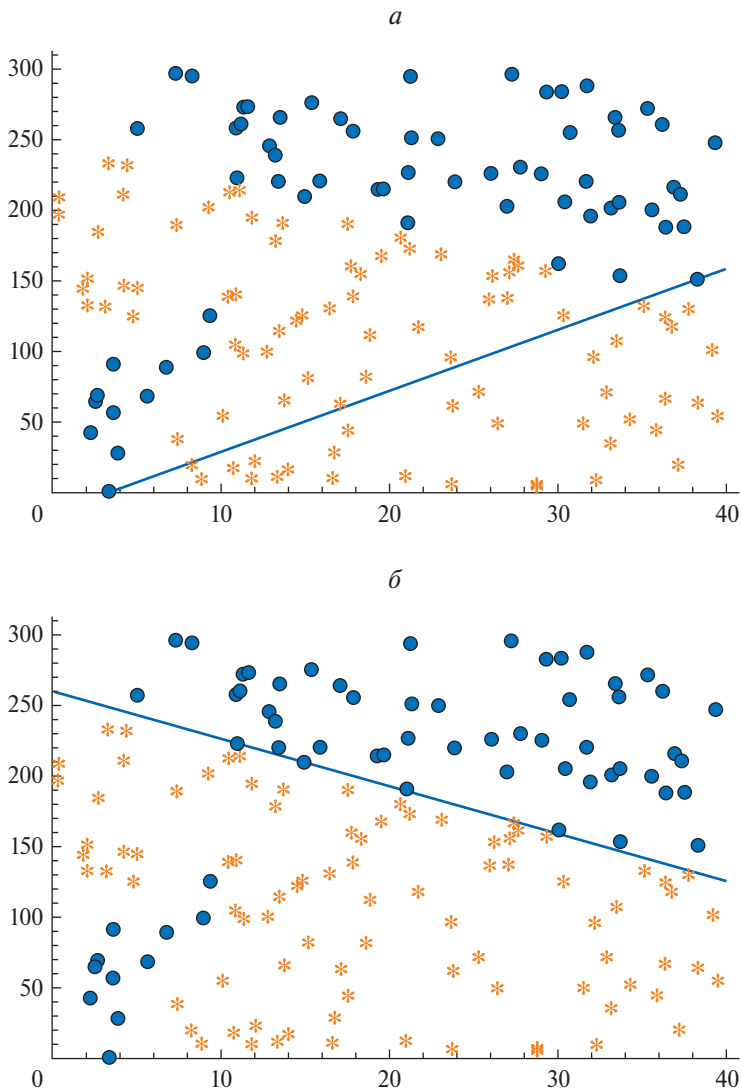


Рис. 21. Проведение разделяющей линии:
 а) $y_1 = 260,58 - 3,3722x$; б) $y_1 = -14,01 + 4,3135x$

На второй итерации также сперва сравниваем два варианта прямой y_2 (рис. 23).

Согласно рис. 23, для первого уравнения количество нарушений условий равно 3. Если воспользоваться данным решающим правилом, то останется классифицировать 91 элемент ($88 + 3$). В свою

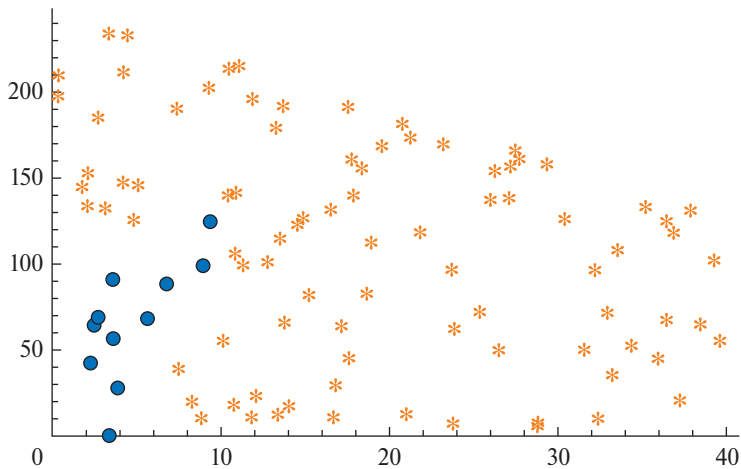


Рис. 22. Множества после отсечения на первой итерации

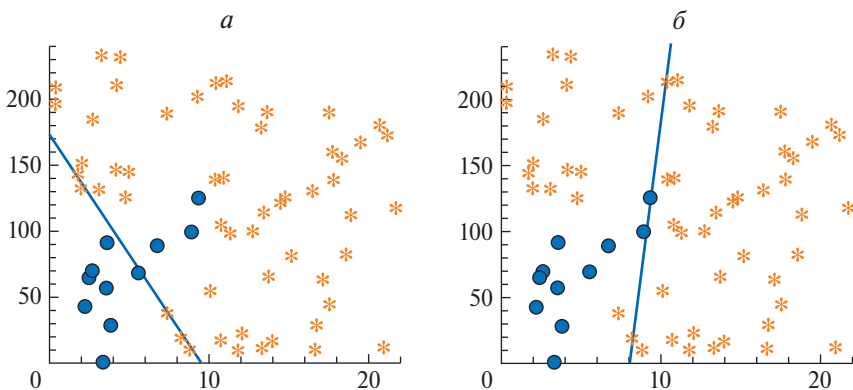


Рис. 23. Выбор разделяющей линии y_2 :

а) $y_2 = 173,26 - 18,197x$; б) $y_2 = -762,3 + 94,263x$

очередь, для второго уравнения количество нарушений равно 16. Если воспользоваться данным решающим правилом, то останется классифицировать 37 элементов (11 + 16).

Выбираем множество с наименьшим количеством оставшихся элементов. Количество оставшихся элементов множества МР2 равно 16. Отсечем полученное на второй итерации множество и получим изображения, как на рис. 24.

На третьей итерации проведем два варианта разделяющей линии y_3 (рис. 25).

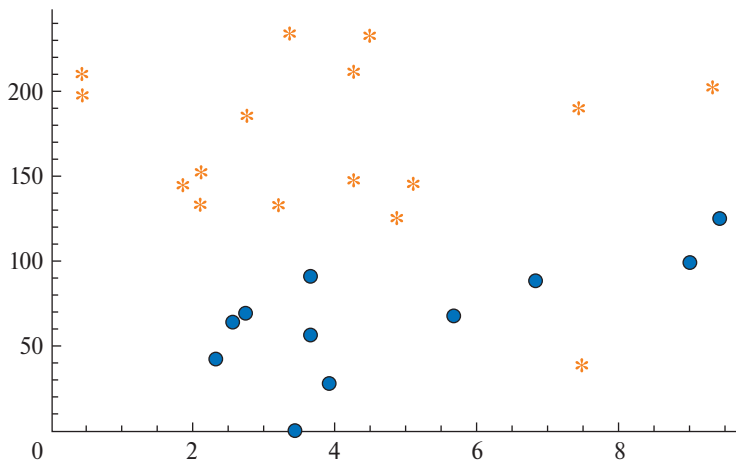


Рис. 24. Множества после отсечения на второй итерации

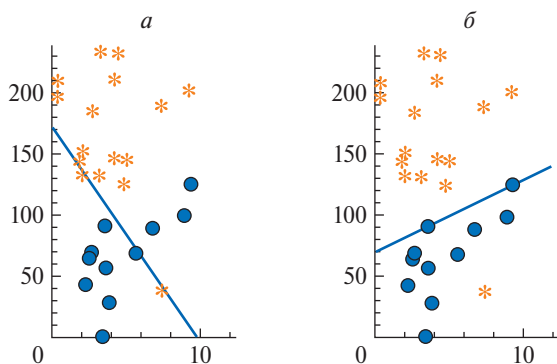


Рис. 25. Выбор разделяющей линии y_3 :
 а) $y_3 = 172,01 - 17,6x$; б) $y_3 = 70,155 + 5,9061x$

Как видно из уравнения на рис. 25, а количество нарушений условий равно 3. Если воспользоваться данным решающим правилом, то останется классифицировать 19 элементов (3 + 16). Для второго уравнения количество нарушений условий равно 1. Если воспользоваться данным решающим правилом, то останется классифицировать 12 элементов (1 + 11).

Выбираем правило с наименьшим количеством оставшихся элементов. На рис. 26 показаны полученные множества после отсечения. Видно, что множества линейно разделимы, поэтому для

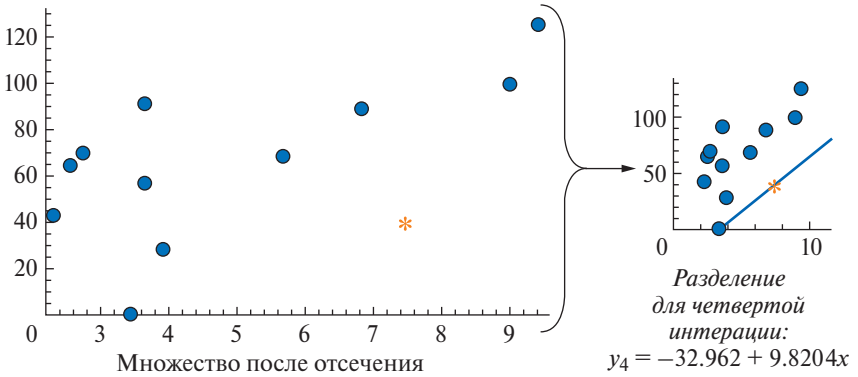


Рис. 26. Отсечение множества на третьей итерации и разделение на четвертой

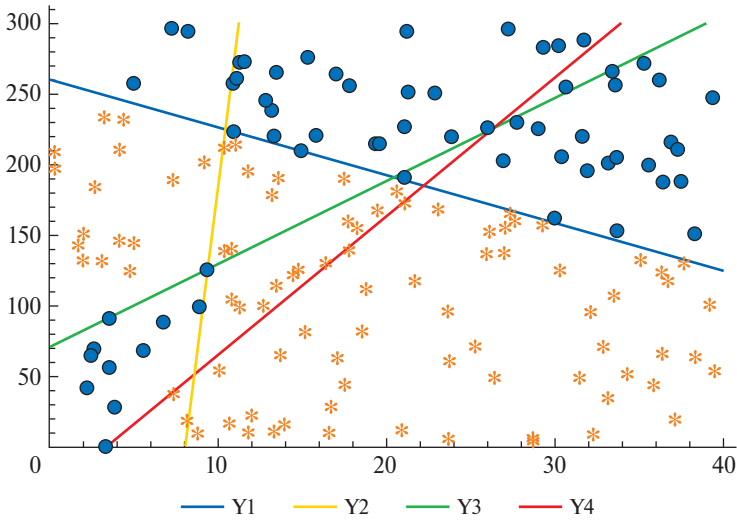


Рис. 27. Разделение множеств четырьмя линейными классификаторами

четвертой итерации нет необходимости сравнивать различные уравнения прямой y_4 .

Соответственно общий рисунок полученной классификации множеств можно представить в соответствии с изображением на рис. 27.

Формулировка КС в данном случае будет следующая:

1. Если точка расположена выше линии y_1 , то она относится к множеству MP_1 , в противном случае решение принимает следующий по рангу член.

2. Если точка расположена ниже линии y_2 , то она относится к множеству MP_2 , в противном случае решение принимает следующий по рангу член.

3. Если точка расположена выше линии y_3 , то она относится к множеству MP_2 , в противном случае решение принимает следующий по рангу член.

4. Если точка расположена выше линии y_4 , то она относится к множеству MP_1 , в противном случае она относится к множеству MP_2 .

В данном случае на каждой итерации решалось две задачи, т.е. условия строгости и жесткости задавались поочередно для каждого из множеств. Для другого множества искалось решающее правило (гиперплоскость), обеспечивающее оптимальное отсечение. В качестве члена комитета выбиралось правило, после применения которого оставалось минимальное суммарное количество элементов множеств, подлежащих классификации. Уменьшение количества элементов, подлежащих классификации на каждом шаге, обеспечивает сходимость алгоритма, но в ряде случаев требуется большое количество итераций и комитет получается излишне громоздким.

Естественным образом возникает вопрос, обязательно ли осуществлять действия последовательно и можно ли построить математические модели, где члены комитета ищутся одновременно в рамках одной модели. Ответ смотрите далее.

3.3. МОДЕЛЬ КОМИТЕТА ЕДИНОГЛАСИЯ

Рассмотрение вопроса, как можно построить единую модель комитета, мы начнем с примера для КЕ, построенного на рисунках, приведенных в предыдущем параграфе. Дополним множество обозначений:

T – множество членов комитета (гиперплоскостей);

t – индекс множества T ;

p_{ij} – параметры наблюдения j ;

a_i^t – коэффициенты гиперплоскостей (искомые переменные);

b^t – свободные члены гиперплоскостей (искомые величины);

z_j^t – булевы переменные для фиксации нарушений условий разделения множеств;

h – число членов комитета (гиперплоскостей, нейронов).

Напомним, что в случае КЕ за элементы одного из множеств все члены комитета голосуют единогласно, а за элементы другого множества — хотя бы один член против. Поэтому по аналогии с итерационным методом построения КС на начальном этапе для одного из множеств условия зададим жестко и строго. В символьном виде это можно записать как КЕ с жесткими условиями для одного из множеств:

$$\sum_{i \in I} p_{ij} * a_i^t + b^t \leq -\varepsilon, \quad j \in J_1, \quad t \in T \quad (5.1)$$

$$\sum_{i \in I} p_{ij} * a_i^t + b^t + L * z_j^t \geq \varepsilon, \quad j \in J_2, \quad t \in T \quad (5.2)$$

$$\sum_{t \in T} z_j^t \leq 0, \quad j \in J_1 \quad (5.3)$$

$$\sum_{t \in T} z_j^t \leq h - 1, \quad j \in J_2 \quad (5.4)$$

$$\min \sum_{t \in T} \sum_{j \in J_2} z_j^t \quad (5.5)$$

Решение:

— первый член комитета: $a_1 = 0,88895$; $a_2 = 1$; $b = -182,16$.

— второй член комитета: $a_1 = 5,5526$; $a_2 = 1$; $b = -322,93$

В общем случае нам неизвестно, для какого из множеств условия должны быть записаны без корректировок. Попробуем построить модель, где такой выбор осуществляется автоматически в виде КЕ с выбором множества с жесткими условиями:

$$\sum_{i \in I} p_{ij} * a_i^t + b^t - L * z_j^t \leq -\varepsilon, \quad j \in J_1, \quad t \in T \quad (5.6)$$

$$\sum_{i \in I} p_{ij} * a_i^t + b^t + L * z_j^t \geq \varepsilon, \quad j \in J_2, \quad t \in T \quad (5.7)$$

$$\sum_{t \in T} z_j^t \leq (h - 1) * Z, \quad j \in J_1 \quad (5.8)$$

$$\sum_{t \in T} z_j^t \leq (h - 1) * (1 - Z), \quad j \in J_2 \quad (5.9)$$

$$\min \sum_{t \in T} \sum_{j \in J} z_j^t, \quad (5.10)$$

где Z — булева переменная для выбора множества (0 — условия единогласия выполняются для множества J_1 ; 1 — условия единогласия выполняются для множества J_2).

Отметим, что, начиная с этого примера, мы не будем специально оговаривать условие строгости для ограничений, полагая, что в случае необходимости добавить малые величины в правые части ограничений не составит труда.

Решение:

– первый член комитета: $a_1 = 5,5518$; $a_2 = 1$; $b = -322,89$.

– второй член комитета: $a_1 = 0,90838$; $a_2 = 1$; $b = -183,88$.

Некоторое различие в коэффициентах по сравнению с предыдущим решением не должно смущать исследователя, так как мы уже поясняли, что это не угроза, а возможность, и в дальнейшем на таких различиях останавливаться не будем.

Итоги голосования можно интерпретировать следующим образом:

1. Для точек множества МР1 оба члена комитета не возражали за отнесение их к множеству МР1.

2. Для точек множества МР2 хотя бы один член комитета возражал за отнесение их к множеству МР2.

Приведем другие примеры КЕ. Например, указанные множества можно разделить КЕ из следующих трех членов:

$$a_2 = 200 - 7a_1; a_2 = 47 + 5a_1; a_2 = 10 + 17a_3 \text{ (рис. 28).}$$

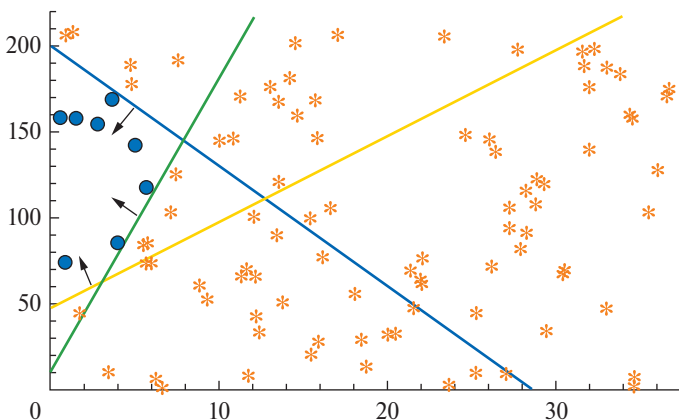


Рис. 28. Первый пример разделения множеств КЕ

Указанные множества можно разделить КЕ из следующих трех членов:

$$y_1 = 300 - 7a; y_2 = 67 + 5a; y_3 = 10 + 17a \text{ (рис. 29).}$$

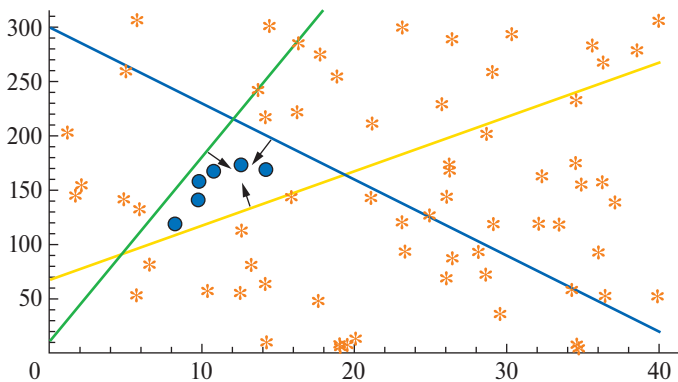


Рис. 29. Второй пример разделения множеств КЕ

3.4. МОДЕЛЬ КОМИТЕТА БОЛЬШИНСТВА

Достаточно часто разделить множества КЕ не представляется возможным. Графически это может выглядеть, как показано на рис. 30.

Очевидно, что линейно или КЕ эти множества не разграничить. Но для практики рекомендуем попробовать. Воспользуемся опять идеологией комитетного подхода. Пусть голоса членов комите-

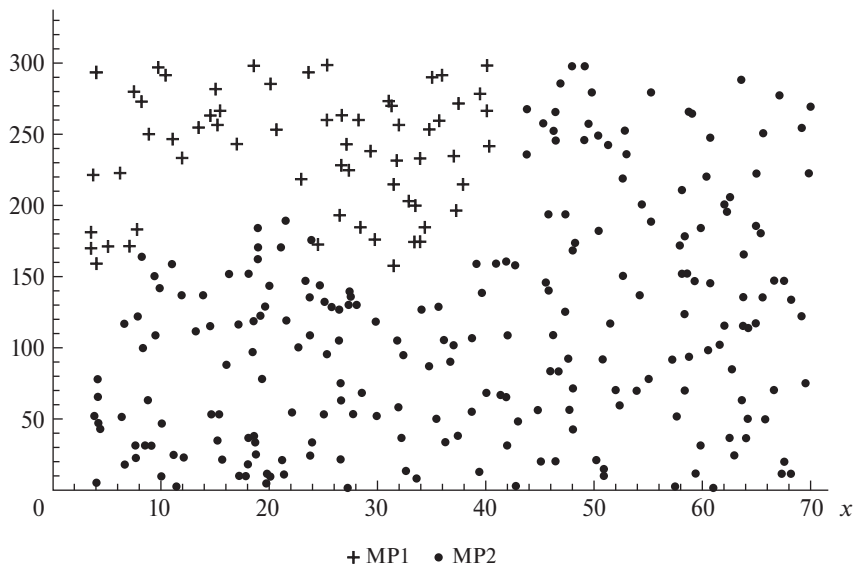


Рис. 30. Пример множеств неразделимых КЕ

та равнозначны, а количество членов (гиперплоскостей) нечетно. Необходимо найти такой набор гиперплоскостей, чтобы для каждой точки множества МР1 простое большинство (больше половины) из них голосовало «за», а для каждой точки множества МР2 — «против».

Математически это можно записать как КБ, заменив в модели КЕ с выбором множества с жесткими условиями ограничения (5.8) и (5.9) на (6.1) и (6.2) соответственно:

$$\sum_{i \in T} z_j^i \leq m, \quad j \in J_1, \quad (6.1)$$

$$\sum_{i \in T} z_j^i \leq h - m - 1, \quad j \in J_2, \quad (6.2)$$

где m — меньшинство (задаваемая константа), h — количество членов комитета.

Решение:

— первый член комитета: $a_1 = 3,36119$; $a_2 = -1$; $b = 136,92$;

— второй член комитета: $a_1 = 11,2154$; $a_2 = -1$; $b = -223,1$;

— третий член комитета: $a_1 = -5,49795$; $a_2 = -1$; $b = 308,5$.

В данном примере мы задали меньшинство как константу $m = 1$. Квалифицированное меньшинство может быть вычисляемо. Для случая трех членов это не имеет смысла, а, например, при 9 членах оно может быть как 5, так и 7. Эти рассуждения понадобятся нам в дальнейшем.

Наверное, глядя на эту модель, хочется сказать: «Да, это скорее комитет меньшинства». Поэтому поясним, что приводимый нами способ подсчета голосов и форма записи условий комитета не являются единственными. По сути, мы математически записали, что, если член комитета согласен, то он воздерживается ($z_j^i = 0$), а если не согласен, то голосует против ($z_j^i = 1$). Иногда кажется, что более естественным будет сделать наоборот. Это легко достигается введением дополнительного условия $\alpha_j^i = 1 - z_j^i$ $j \in J$, а условия комитета принимают вид

$$\sum_{i \in T} \alpha_j^i \geq \beta, \quad j \in J_1, \quad (6.3)$$

$$\sum_{i \in T} \alpha_j^i \geq \beta, \quad j \in J_2, \quad (6.4)$$

где β — большинство, т.е. более половины.

Конечно, можно решать задачу и в таком виде. Более того, можно даже согласиться, что так она более интерпретируема,

но проигрывает по числу переменных, а значит, на задачах большой размерности и по времени счета. Кроме того, если вы сделаете подстановку в условия (6.3) и (6.4), то получите условия (6.1) и (6.2).

Отметим также, что таким же естественным является подход, когда «за» это 1, а «против» -1 . Достигается это тоже довольно легко, введением дополнительного условия $h_j^t = 1 - 2 * z_j^t, j \in J$, а как записать условия комитетов, предлагаем сообразить самостоятельно. Мы не оговорились, именно комитетов, т.е. и для единогласия, и всех последующих. Поэтому каждый должен определиться, какая форма записи ему ближе, и в дальнейшем использовать ее.

Приведем различные примеры комитетов большинства.

Посмотрим на рис. 31 множества МР1 (68 элементов) и МР2 (100 элементов).

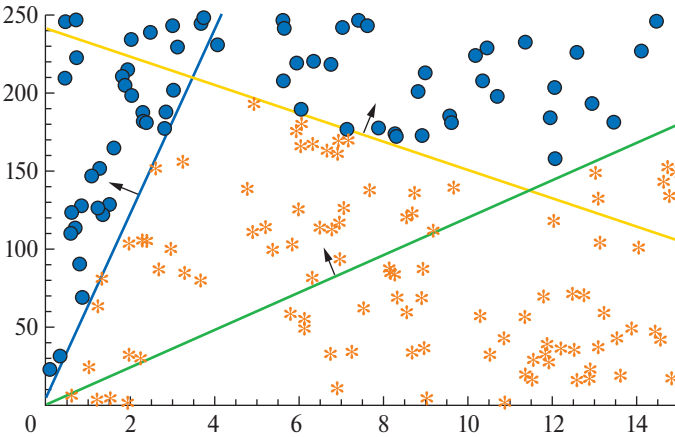


Рис. 31. Первый пример разделения множеств КБ

Данные множества можно разделить комитетом большинства, состоящим из трех членов: $y_1 = 5 + 59a$; $y_2 = 241 - 9a$; $y_3 = 12a$ (рис. 31).

Данные множества можно разделить КБ из трех членов: $y_1 = 200 - 13a$; $y_2 = 47 + 5a$; $y_3 = -100 + 17a$ (рис. 32).

Данные множества можно разделить комитетом большинства, состоящим из 5 членов: $y_1 = 5 + 59a$; $y_2 = 57 + 2a$; $y_3 = -7 + 23a$; $y_4 = 241 - 9a$; $y_5 = 12a$ (рис. 33).

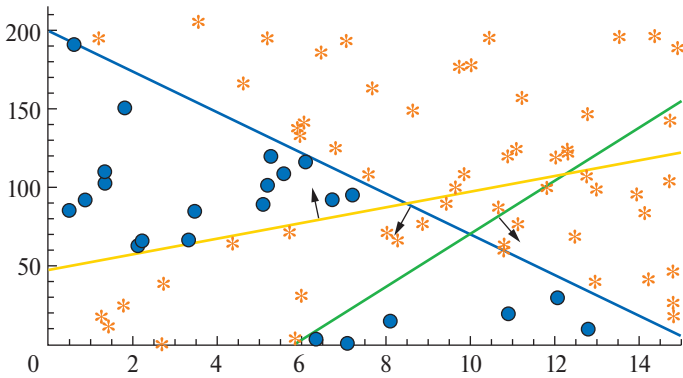


Рис. 32. Второй пример разделения множеств КБ

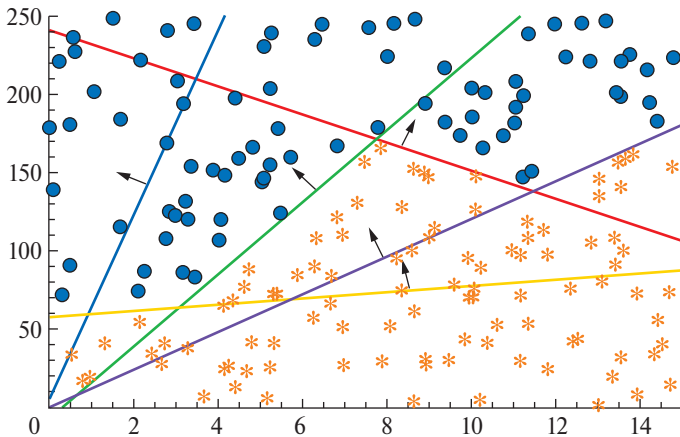


Рис. 33. Третий пример разделения множеств КБ

Надеемся, что пытливым читатель заметит, что в данном примере одна из линий лишняя, исключит ее и проинтерпретирует это, используя идеологию метода комитетов.

Ответ: это КБ из четырех членов с квалифицированным большинством 3.

Данные множества можно разделить КБ, состоящим из пяти членов:

$y_1 = 5 + 59a$; $y_2 = 97 + 2a$; $y_3 = -7 + 23a$; $y_4 = 241 - 9a$; $y_5 = 12a$ (рис. 34).

Рис. 34 очень похож на предыдущий, те же линии, но точки расположены чуть по-другому.

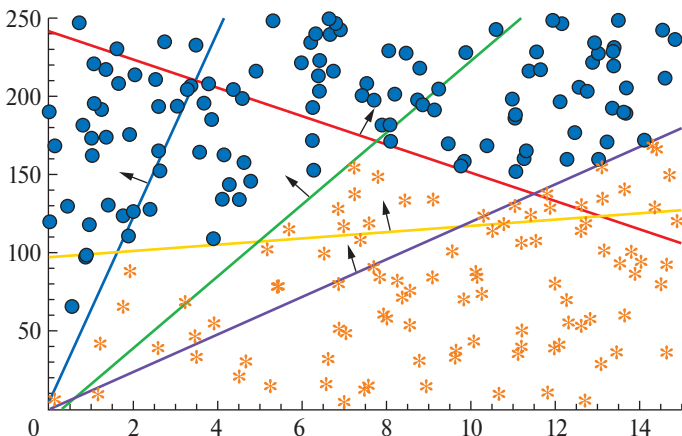


Рис. 34. Четвертый пример разделения множеств КБ

3.5. МОДЕЛЬ КОМИТЕТА СТАРШИНСТВА

До сих пор при разделении множеств гиперплоскостями и интерпретируя их как членов комитета, голосующих за или против, мы исходили из того, что голоса всех членов комитета имеют одинаковый вес. Для иллюстрации подхода и сравнительного анализа воспользуемся множествами из примеров, которые мы использовали при описании итерационного подхода.

Отметим, что, следуя данному правилу, после получения ответа «да» на любом шаге можно принимать решение. Это напоминает комитет, где вес голоса старшего члена больше всех остальных вместе взятых. Если он воздержался, то решение принимает следующий по старшинству и так далее. Поэтому такие комитеты называются комитетами старшинства.

Для составления КС с фиксированными весами достаточно в предыдущей модели преобразовать (6.1) и (6.2) в (7.1) и (7.2) соответственно и поменять целевую функцию:

$$\sum_{t \in T} z_j^t * V^t \leq m, \quad j \in J_1, \quad (7.1)$$

$$\sum_{t \in T} (z_j^t * V^t) \leq \sum_{t \in T} V^t - m - 1, \quad j \in J_2, \quad (7.2)$$

$$\min m$$

где V^t – веса членов комитета (заранее задаваемые величины по принципу вес старшего больше суммы всех остальных, для этого

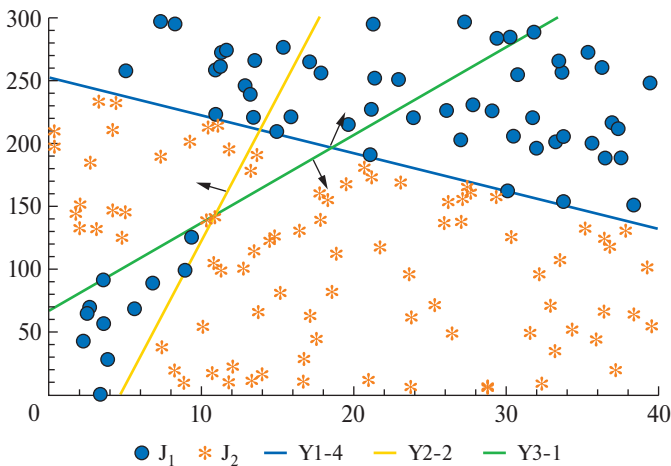


Рис. 35. Первый пример разделения множеств КС

удобно использовать степени 2 или 10). Таким образом можно оцифровать все сектора; m – ограничитель по весу (переменная).

Попробуем разделить КС множества, используемые нами при построении разделения итерационным методом (рис. 20). Для этого сперва присвоим каждому члену комитета веса, равные степеням 2, то есть 4, 2 и 1. В этом случае решением будет $y_1 = 251 - 3a$ ($V = 4$); $y_2 = -51 + 15a$ ($V = 2$); $y_3 = 122 + a$ ($V = 1$) (рис. 35).

Чтобы понять, как указанное решение классифицируют голубые и желтые точки на рис. 36, последовательно представлено присвоение весов от старшего члена комитета к младшему. Исходя из графика 4 на рис. 36, можно говорить, что объект принадлежит к кружкам, если область, в которой он находится, имеет вес не более 4, иначе он принадлежит к звездочкам.

Теперь, как было обещано, дадим пояснения, почему удобно, чтобы одна из переменных принимала значения 1 или -1 . Запишем уравнения гиперплоскостей из рис. 37 немного в другом виде:

$$1) -3*a_1 - 1*a_2 + 251 = 0;$$

$$2) 15*a_1 - 1*a_2 - 51 = 0;$$

$$3) -1*a_1 + 1*a_2 - 122 = 0.$$

Именно в таком виде наиболее удобно анализировать результаты расчетов в Excel таблицах и определять направление голосования. Гиперплоскости с -1 при p_2 голосуют в одну сторону, те, у которых 1 – в другую. Тогда относительно множества МР1: y_1 и y_2 – «вверх»; y_3 – «вниз». Относительно множества МР2 наоборот.

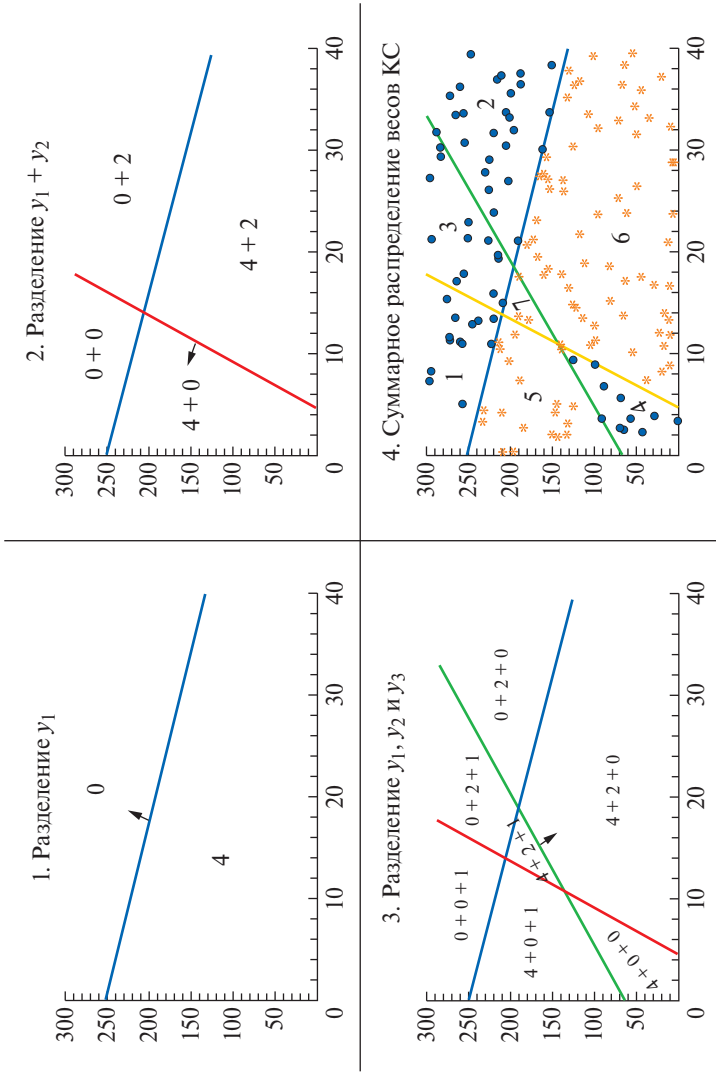


Рис. 36. Логика работы KS для примера с рис. 35

Сравним найденное решение с решением, полученным итерационным способом. Все решения правильно разделяют множества, только при решении в рамках одной задачи понадобилось членов комитета на один меньше. Таким образом, мы видим, что итерационный метод выигрывает по размерности задачи (число переменных и ограничений), а значит и времени счета, но проигрывает по возможности находить минимальный комитет. Минимальный комитет, конечно, не надо фетишизировать и на практике совсем не обязательно использовать именно его, но возможность его находить открывает дополнительные возможности при анализе данных. Так, в данной задаче, если существует комитет из трех членов, то он будет найден.

Приведем различные примеры КС.

Данные множества можно разделить КС, состоящим из 4 членов при $m = 10$: 1) $a_2 = 101 + 13a_1$ ($V = 8$); 2) $a_2 = 57 + 7a_1$ ($V = 4$); 3) $a_2 = 13 + 5a_1$ ($V = 1$); 4) $a_2 = -51 + 11a_1$ ($V = 2$) (рис. 37).

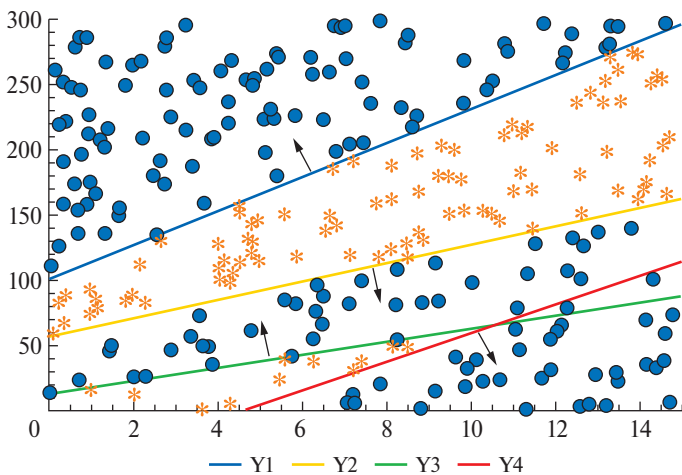


Рис. 37. Второй пример разделения множеств КС

Данные множества можно разделить КС, состоящим из 5 членов при $m = 28$: 1) $a_2 = -10a_1 + 266$ ($V = 16$); 2) $a_2 = 13a_1 + 101$ ($V = 8$); 3) $a_2 = 7a_1 + 57$ ($V = 2$); 4) $a_2 = 5a_1 + 13$ ($V = 1$); 5) $a_2 = 14a_1 - 74$ ($V = 4$) (рис. 38).

Чуть выше, до приведения примеров, мы обещали рассмотреть случай, когда комитет из заданного числа членов не существует. То есть задача противоречива, и вместо конкретного решения вы получите ответ «значения отсутствуют». Означает ли это, что ситуация тупиковая? Конечно, нет. Надо просто условия для комитета

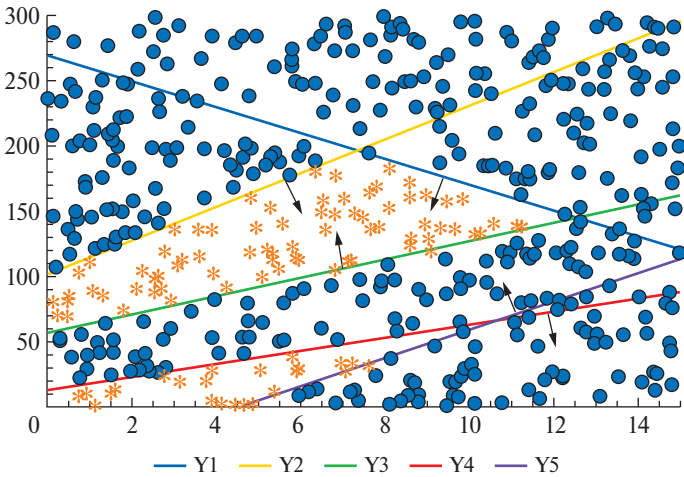


Рис. 38. Третий пример разделения множеств КС

записать в нежестком виде и потребовать минимизации их нарушений. Составим такую модель КС с фиксированными весами и корректировкой условий комитета в случае противоречивости условий задачи, преобразовав в предыдущей модели (7.1), (7.2) в (7.3) и (7.4) соответственно:

$$\sum_{t \in T} z_j^t * V^t \leq m + L * E_j, \quad j \in J_1, \quad (7.3)$$

$$\sum_{t \in T} (z_j^t * V^t) \leq \sum_{t \in T} V^t - m - 1 + L * E_j, \quad j \in J_2, \quad (7.4)$$

$$\min \sum_{j \in J} E_j, \quad (7.5)$$

где V^t – веса членов комитета (заранее задаваемые величины по принципу вес старшего больше суммы всех остальных, удобно использовать степени 2);

m – меньшинство (переменная);

E_j – булевы переменные для фиксации нарушений классификации.

Решая задачу в таком виде в случае непротиворечивости условий, вы получаете допустимое решение, в случае противоречивости – оптимальную коррекцию условий. А далее выбор за вами:

1. Согласиться с коррекцией, если нарушений немного.
2. Увеличить число членов комитета.
3. Попробовать модель с автоматическим подбором весов.

Дополнительно анализируя модель КС, можно прийти к интересному выводу, что указанная модель также подходит для КЕ и КБ. Для доказательства обозначенного вывода сначала вспомним, какие ограничения используются в каждой из моделей комитета (табл. 2).

Таблица 2

Ограничения в моделях комитетов

Логика	$j \in J_1$	$j \in J_2$
КЕ	$\sum_{t \in T} z_j^t \leq 0 \quad (5.3)$	$\sum_{t \in T} z_j^t \leq h - 1 \quad (5.4)$
КБ	$\sum_{t \in T} z_j^t \leq m \quad (6.1)$	$\sum_{t \in T} z_j^t \leq h - m - 1 \quad (6.2)$
КС	$\sum_{t \in T} (z_j^t * V^t) \leq m \quad (7.1)$	$\sum_{t \in T} (z_j^t * V^t) \leq \sum_{t \in T} V^t - m - 1 \quad (7.2)$

Рассмотрим КС при $V^t = 1 \quad \forall t \in T$:

– если $m = 0$ или $h - 1$, то (7.1) и (7.2) могут быть соответственно преобразованы в (5.3) и (5.4), что соответствует КЕ;

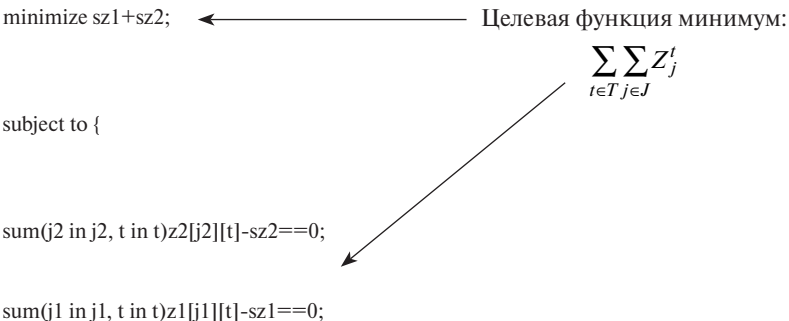
– если $0 < m < h - 1$, то (7.1) и (7.2) могут быть соответственно преобразованы в (6.1) и (6.2), что соответствует КБ.

Рассмотрим КС при $V^t = 2^t$:

– если $m = 0$ или $\sum_{t \in T} V^t - 1$, то (7.1) и (7.2) могут быть соответственно преобразованы в (5.3) и (5.4), что соответствует КЕ;

– если $0 < m < \sum_{t \in T} V^t - 1$, то (7.1) и (7.2) не могут быть преобразованы, значит КС.

Для лучшего усвоения материала приведем схему соответствия модели КС (без корректировок условий комитета) ее программному коду:



$$\sum_{i \in I} (p_{ij} * a_i^t) + b^t - L * z_j^t \leq -\varepsilon \quad j \in J_1, \quad t \in T$$

forall(t in T)forall(j1 in J1)sum(i in I)MP1[j1][i]*a[i][t]+b[t]<=L*z1[j1][t]-0.01;

$$\sum_{i \in I} (p_{ij} * a_i^t) + b^t + L * z_j^t \geq \varepsilon \quad j \in J_2, \quad t \in T$$

forall(t in T)forall(j2 in J2)sum(i in I)MP2[j2][i]*a[i][t]+b[t]>=L*z2[j2][t]+0.01;

$$\sum_{t \in T} (z_j^t * V^t) \leq m \quad j \in J_1$$

forall(j1 in J1)sum(t in T)V[t]*z1[j1][t]<=m;

$$\sum_{t \in T} (z_j^t * V^t) \leq \sum_{t \in T} V^t - m - 1 \quad j \in J_2$$

forall(j2 in J2)sum(t in T)V[t]*z2[j2][t]<=sv-m-1;

}

В данном примере $sv =$ сумме весов, т.е. $\sum_{t \in T} V^t$.

Данной константе надо просто присвоить конкретное значение в зависимости от числа членов комитета.

3.6. МОДЕЛЬ КОМИТЕТА С ПОДБОРОМ ВЕСОВ

В предыдущих примерах веса членов комитета задавались заранее. Естественным образом возникает идея, давайте и V^t определим как переменные и решим задачу. Вся беда в том, что в этом случае условия (7.1) и (7.2) будут содержать произведение двух переменных z_j^t и V^t . А в таком виде задача не решаема. Попробуем все же воспользоваться тем, что одна из переменных булева. Введем дополнительные переменные v_j^t произведения $z_j^t * V^t$. Нам необходимо добиться, чтобы $v_j^t = 0$, если $z_j^t = 0$ и $v_j^t = V^t$, если $z_j^t = 1$.

Выполнение этих условий достигается в рамках следующей модели:

Комитет с автоматическим подбором весов (далее – КАПВ) и корректировкой условий комитета в случае противоречивости условий задачи:

$$\sum_{i \in I} p_{ij} * a_i^t + b^t - L * z_j^t \leq -\varepsilon, \quad j \in J_1, t \in T, \quad (8.1)$$

$$\sum_{i \in I} p_{ij} * a_i^t + b^t + L * z_j^t \geq \varepsilon, \quad j \in J_2, t \in T, \quad (8.2)$$

$$\sum_{t \in T} V^t = 1, \quad (8.3)$$

$$v_j^t \geq V^t + z_j^t - 1, \quad j \in J, t \in T, \quad (8.4)$$

$$0 \leq v_j^t \leq z_j^t, \quad j \in J, t \in T, \quad (8.5)$$

$$v_j^t \leq V_j^t, \quad j \in J, t \in T, \quad (8.6)$$

$$\sum_{t \in T} v_j^t * h \leq m + L * E_j, \quad j \in J_1, \quad (8.7)$$

$$\sum_{t \in T} v_j^t * h \leq h - m - \varepsilon + L * E_j, \quad j \in J_2, \quad (8.8)$$

$$m \leq h - 1, \quad (8.9)$$

$$\min \sum_{j \in J} E_j, \quad (8.10)$$

где V^t – веса членов комитета, вычисляемые в ходе решения задачи; v_j^t – вспомогательная переменная; m – квалифицированное меньшинство при весах V^t .

Обращаем внимание, что в условиях (8.7), (8.8), (8.9) мы используем константу h – число членов комитета. Очевидным образом от ее использования можно отказаться. Апробируем данную модель на данных, отображенных ранее на рис. 20. Напомним, что начальные множества – МР1 (63 элемента) и МР2 (88 элементов).

Соответственно был найден комитет без нарушения условий комитета со следующими гиперплоскостями и их весами (рис. 39):

$$v_1 = 0,333; v_2 = 0,66667; v_3 = \text{остаток}$$

$$y_1 = -181,5 + 29,691a; y_2 = 248,47 - 2,86a; y_3 = 66,166 + 6,9968a$$

Давайте добавим точку [30, 100] и посмотрим, как будет вести себя задача в этом случае. Несмотря на то что задача стала проти-

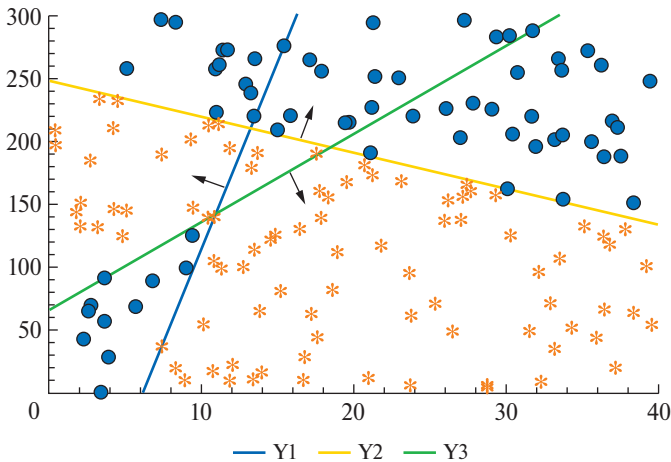


Рис. 39. Пример КАПВ в случае отсутствия противоречий

воречивой, найден комитет с одним нарушением условий комитета и следующими гиперплоскостями, и их весами (рис. 40):

$$v_1 = 0,333; v_2 = 0,333; v_3 = 0,3333$$

$$y_1 = 257,79 - 3,1695a; y_2 = 128,32 - 0,268a; y_3 = -181,5 + 29,691a$$

Усложним задачу еще – добавим точку [22, 150]. В результате решения будет найден комитет с двумя нарушениями условий комитета и следующими гиперплоскостями, и их весами (рис. 41):

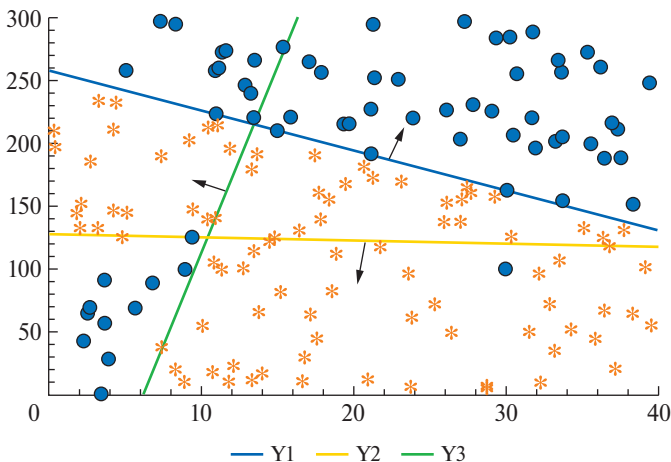


Рис. 40. Пример КАПВ в случае первого противоречия для трех членов

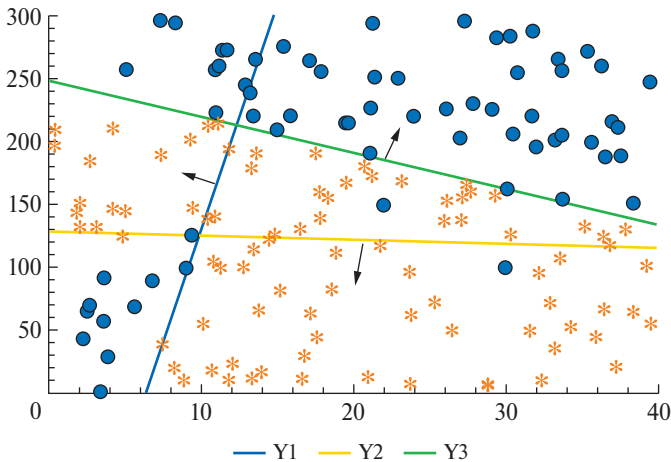


Рис. 41. Пример КАПВ в случае двух противоречий для трех членов

$$v_1 = 0,333; v_2 = 0,000333; v_3 = 0,66666$$

$$y_1 = -228,32 + 35,955a; y_2 = 129,12 - 0,35234a; y_3 = 248,47 - 2,86a$$

Теперь попробуем для последнего примера сформировать комитет из четырех членов. В результате решения будет найден комитет с одним нарушением ($E1(2) = 1$) условий комитета и следующими гиперплоскостями, и их весами (рис. 42):

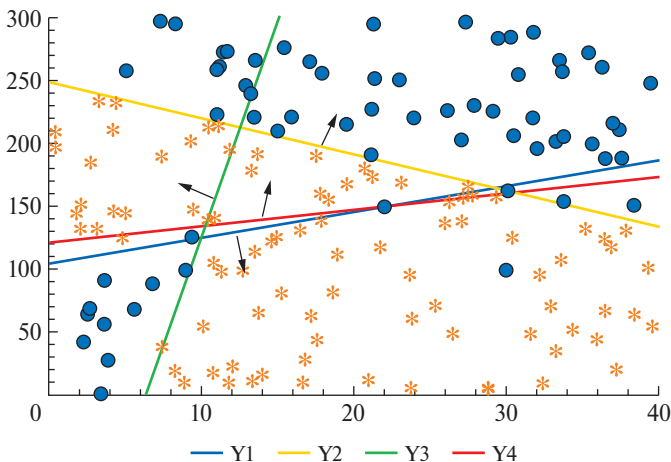


Рис. 42. Пример КАПВ в случае двух противоречий для четырех членов

$$v_1 = 0,00025; v_2 = 0,5005; v_3 = 0,2495; v_4 = 0,24975$$

$$y_1 = 105,07 + 2,0421a; y_2 = 248,47 - 2,86a; y_3 = -215,82 + 34,282a;$$

$$y_4 = 120,98 + 1,3191a$$

Посмотрим, какое решение будет в случае комитета из пяти членов. В этом случае нарушений условий комитета не будет, а гиперплоскости и их веса будут следующие (рис. 43):

$$v_1 = 0,0002; v_2 = 0,7998; v_3 = 0,0004; v_4 = 0,1994; v_5 = 0,0002$$

$$y_1 = -215,82 + 34,282a; y_2 = 248,47 - 2,86a; y_3 = 55,292 + 9,9691a;$$

$$y_4 = 292,26 - 6,2486a; y_5 = 292,02 - 6,6269a$$

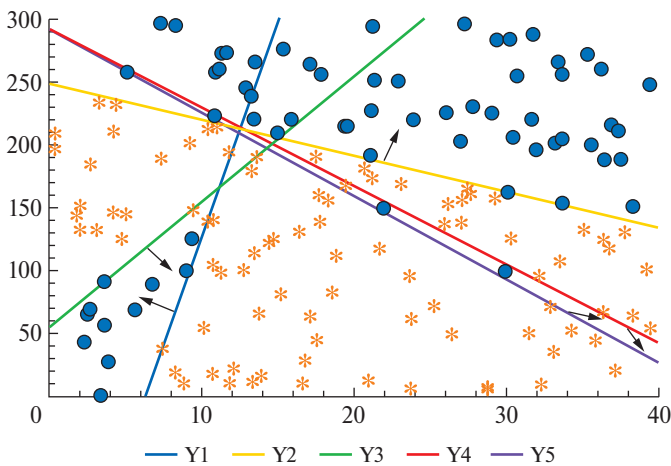


Рис. 43. Пример КАПВ в случае двух противоречий для пяти членов

Теперь проверим комитет с автоматическим подбором весов на данных КЕ. Начальные множества МР1 (6 элементов) и МР2 (94 элемента) отображены на рис. 44.

В данном случае будет найден комитет без нарушения условий комитета и следующими гиперплоскостями, и их весами при $m = 0$ (рис. 45):

$$v_1 = 0,99933; v_2 = 0,000333; v_3 = \text{остаток}$$

$$a_2 = -693,42 + 59,634a_1;$$

$$a_2 = 92,506 + 2,5256a_1; a_2 = 88,866 + 7,6529a_1$$

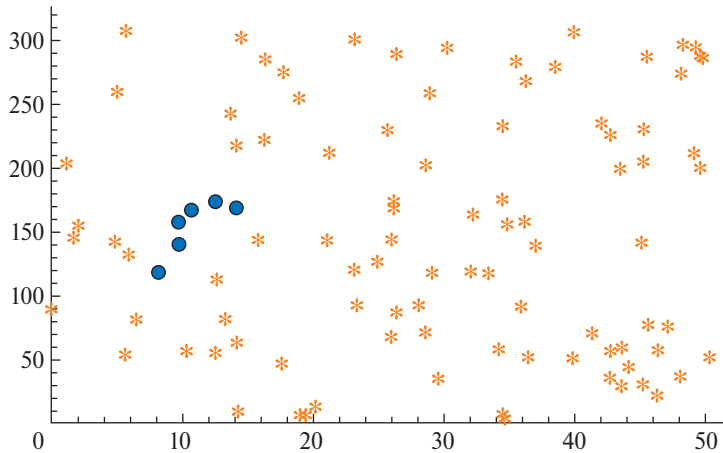


Рис. 44. Множество, разделимое КЕ

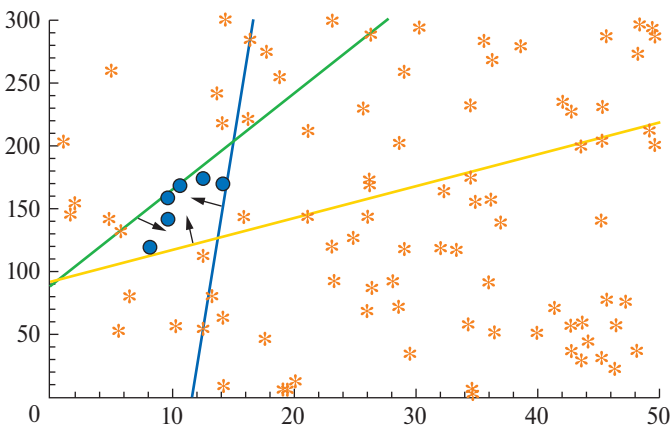


Рис. 45. Пример КАПВ в случае КЕ для трех членов

Теперь проверим КАПВ на данных КБ на еще одной группе множеств. Начальные множества МР1 (68 элементов) и МР2 (100 элемента) отображены на рис. 46.

В данном случае будет найден комитет без нарушения условий комитета и следующими гиперплоскостями, и их весами при $m = 2$ (рис. 47):

$$v_1 = 0,00033; v_2 = 0,333; v_3 = 0,6666$$

$$a_2 = 244,93 - 10,117a_1; a_2 = 177,44 - 1,5658a_1; a_2 = 11,846 + 53,85a_1$$

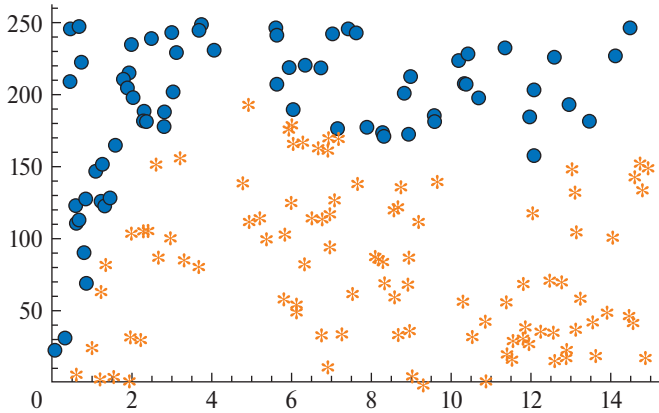


Рис. 46. Множество, разделимое КБ

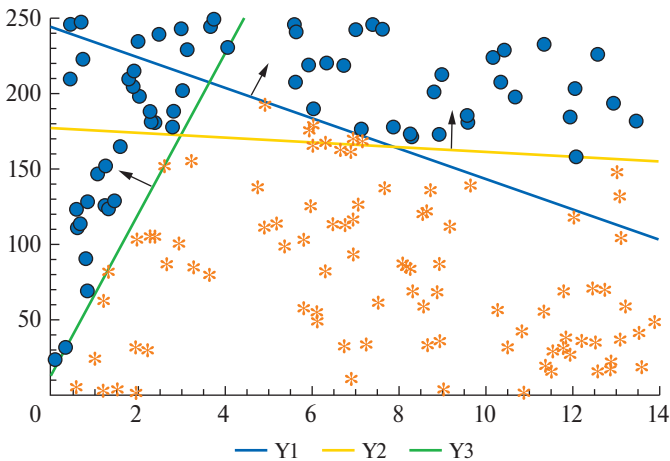


Рис. 47. Пример КАПВ в случае КБ для трех членов

Таким образом, мы проверили работоспособность комитета с автоматическим подбором весов. Данный комитет удобен в работе, так как сам определяет значимость каждого члена. К недостаткам, конечно, можно отнести рост числа переменных. Работоспособность данного комитета мы далее продемонстрируем на классической задаче «Ирисы» Фишера. А пока рассмотрим еще одну потенциальную возможность разделения множеств и построения комитетов.

3.7. РАЗДЕЛЕНИЕ МНОЖЕСТВ НЕЛИНЕЙНЫМИ ФУНКЦИЯМИ

Ранее нами были рассмотрены примеры разделения множеств линейными функциями, однако в жизни можно встретить примеры, для которых эффективное разделение достигается при применении нелинейных функций. В качестве примера допустим, что в результате исследований были получены два множества (рис. 48).

Множества линейно не разделимы. Исследователь хочет проверить гипотезу, что множества разделимы функцией: $y = a_1 * x^2 + a_2 * x + b$. Необходимо найти: a_1, a_2, b .

В принципе в данном подходе нет ничего нового для читателя. Сведение нелинейных зависимостей к линейным, мы подробно разбирали, когда рассматривали задачи регрессии. Разделяющий полином имеет вид

$$y = 1,0754 * x^2 - 6,5327 * x + 8,7221.$$

Отметим, что аналогичным образом можно осуществлять разделение множеств полиномами более высоких степеней и любыми сепарабельными функциями. Только надо понимать, что вид функции или функций должен задаваться исследователем, а коэффициенты перед ними определяем в ходе решения.

Хотелось бы также отметить, что работа с нелинейными функциями требует определенного опыта, а попытки линеаризовать несепарабельные функции часто заканчиваются абсурдными результатами.

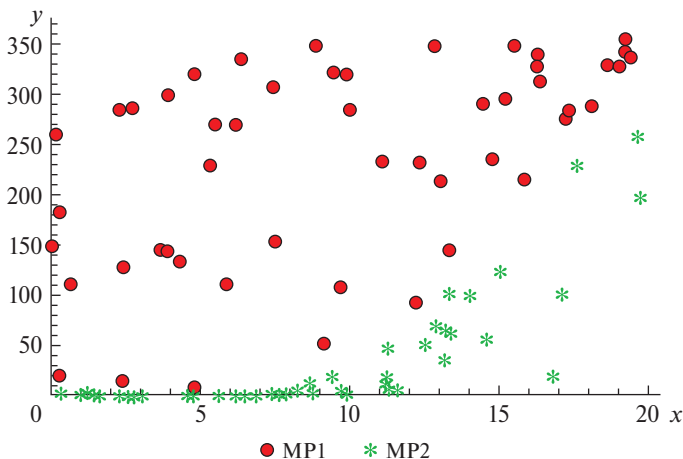


Рис. 48. Пример множества, разделимого нелинейной функцией

3.8. ПРИМЕНЕНИЕ МЕТОДА КОМИТЕТОВ В ИРИСАХ ФИШЕРА И ОЦЕНКЕ ИНФОРМАТИВНОСТИ ПРИЗНАКОВ

Теперь, когда читатель уже научился некоторым методам построения решающих правил, пора перейти к решению реальных задач. Данные будем приводить из общедоступных источников. Наверное, одной из самых ранних и известных является задача Фишера о классификации ирисов. Описание данной задачи и данные читатель может легко найти в интернете, набрав ключевые слова.

В данной задаче рассматриваются не два, а три класса (*setosa*, *virginica*, *versicolor*), причем утверждается, что класс *setosa* линейно отделим от остальных. Поэтому начать отсечение следует с данного класса.

Задачи, содержащие более двух классов, называются многоклассовыми. В общем случае заранее не известно, какое из множеств линейно отделимо и существует или нет линейная разделимость вообще. Но этого и не требуется. Пробуем каждое из множеств вначале отделить от других гиперплоскостью, если не получится, то комитетом. Если одно из множеств или комбинацию множеств (если их больше трех) удалось разделить, то аналогичным образом поступим с разделенными частями.

Продемонстрируем, что ирисы *setosa* можно отделить и комитетом с автоматическим подбором весов, состоящим из одного члена. В нашем случае будет следующее решение: $x_1 = 0,316$; $x_2 = 0$; $x_3 = 0,9$; $x_4 = -1$; $b = 0$.

Сразу обращаем внимание, что все условия выполнены и при этом $x_2 = 0$, значит этот параметр можно исключить из рассмотрения, поэтому о таких параметрах говорят: «Они не информативны».

Прежде чем проверять полученное решение и разделять оставшиеся множества, порассуждаем об информативности признаков на примере ирисов Фишера. Если

- 1) множества линейно делимы,
- 2) параметра четыре,
- 3) один параметр не информативен,
- 4) решение не единственное,

то, может быть, еще какой-то параметр можно исключить из рассмотрения?

Первый пункт означает, что условия разделения можно записать строго и жестко. Второй и третий – что число признаков можно ограничить тремя. Четвертый – что возможна оптимизационная постановка задачи. Преобразуем вышесказанное в модель

определения минимального числа признаков, при линейной разделимости множеств:

$$\sum_{i \in I} p_{ij} * a_i + b \leq -\varepsilon, \quad j \in J_1, \quad (10.1)$$

$$\sum_{i \in I} p_{ij} * a_i + b \geq \varepsilon, \quad j \in J_2, \quad (10.2)$$

$$-L * z_i \leq a_i \leq L * z_i, \quad i \in I, \quad (10.3)$$

$$\min \sum_{i \in I} z_i, \quad (10.4)$$

где z_i – булева переменная для выбора признака (0 – признак не информативен, 1 – признак информативен).

В результате расчетов получаем, на первый взгляд, парадоксальное решение:

$$a_1 = 0; a_2 = 0; a_3 = 1; a_4 = 0; b = -2,99.$$

Это означает, что ирис *Setosa* можно отделить от двух остальных по одному-единственному признаку. Отметим, что на практике обычно так и поступают, просто потому, что так удобней.

Информативность признаков можно определять и при разделении множеств комитетам. То есть, если у всех членов комитета некоторый $a[i] = 0$, то данный признак не информативен. Убедившись, что комитет из определенного числа членов существует, его можно проверить на информативность других признаков.

Модель определения информативности признаков работоспособна, и можете смело использовать ее на более сложных задачах. Недостатком модели (10.1)–(10.4) является ее чувствительность к величине ε . При очень малых величинах ε может происходить потеря точности счета и все a_i будут равны нулю. Поэтому модель может быть дополнена ограничением на сумму z_i .

Однако мы ушли в сторону от задачи построения решающего правила для всего множества Ирисов Фишера.

Исключаем из рассмотрения множество *Setosa* и пробуем разделить *versicolor* и *virginica*. Для практики рекомендуем попробовать различные подходы и начать с попытки линейного деления. Далее мы не будем подробно комментировать каждый шаг. Все они построены на основе модели с автоматическим определением весов членов комитета.

Полностью решающее правило будет таким:

1) если длина внутренней доли околоцветника (от англ. *petal length*) менее 1,9 см или ширина внутренней доли околоцветника (от англ. *petal width*) менее 0,6 см, то это *Setosa*. Согласитесь, что,

используя данное правило, обычный ботаник (в любом смысле) при помощи школьной линейки отделит *Setosa* от других;

2) для дальнейшего разделения множеств использовать один из комитетов на ваш выбор.

Результаты решений приведены в таблицах 3–5.

Таблица 3

**Комитет из двух членов
(вариант 1)**

x1	x2	x3	x4	b	веса
0,06185567	0,28350515	0,36597938	1	2,33405155	0,9995
0,524	-0,452	3,212	1	-19,9158	0,0005

Таблица 4

**Комитет из двух членов
(вариант 2)**

x1	x2	x3	x4	b	веса
0,05426651	0,27132256	0,35659279	1	2,36045426	0,9995
1,399992	-2,199988	4,19998	1	25,4798834	0,0005

Таблица 5

**Комитет из трех членов
(вариант 3)**

x1	x2	x3	x4	b	веса
-0,14036	0,29823	0,368429	1	1,8	0,333
-0,52631	0,39474	2,342078	1	8,7	0,334
0,959991	1,75999	2,439977	1	15	0,333

Обратите внимание, что так как в табл. 5 веса одинаковые, то это КБ. В принципе можно пользоваться любым из приведенных выше комитетов.

Мы достаточно подробно рассмотрели метод комитетов, который позволяет строить решающие правила даже в случае отсутствия линейной разделимости множеств. Достаточно большое число авторов внесли свою лепту в развитие и практическое применение данного подхода [66–72]. На наш взгляд, наиболее системно данный подход развивался в математической школе В.Д. Мазурова [6–8].

ПРИМЕНЕНИЕ ВЫПУКЛЫХ ОБОЛОЧЕК В ЗАДАЧАХ КЛАССИФИКАЦИИ

4.1. ОДНОКЛАССОВАЯ КЛАССИФИКАЦИЯ

Задачи одноклассовой классификации (ЗОК) относятся к типу задач обучения без учителя. ЗОК возникают в случаях, когда по различным причинам к обучению может быть предъявлен только один класс объектов. Такого рода задачи возникают, когда предъявление других классов содержательно не имеет смысла или слишком трудоемко и дорогостояще. Обычно ЗОК связаны с распознаванием индивидуальных признаков (почерка, лица, манеры речи и т.п.) или определением аномалий, таких как:

- Обнаружение подозрительных банковских операций (Credit-card Fraud)

- Обнаружение вторжений (Intrusion Detection)

- Обнаружение нестандартных игроков на бирже (инсайдеров)

- Обнаружение неполадок в механизмах по показаниям датчиков

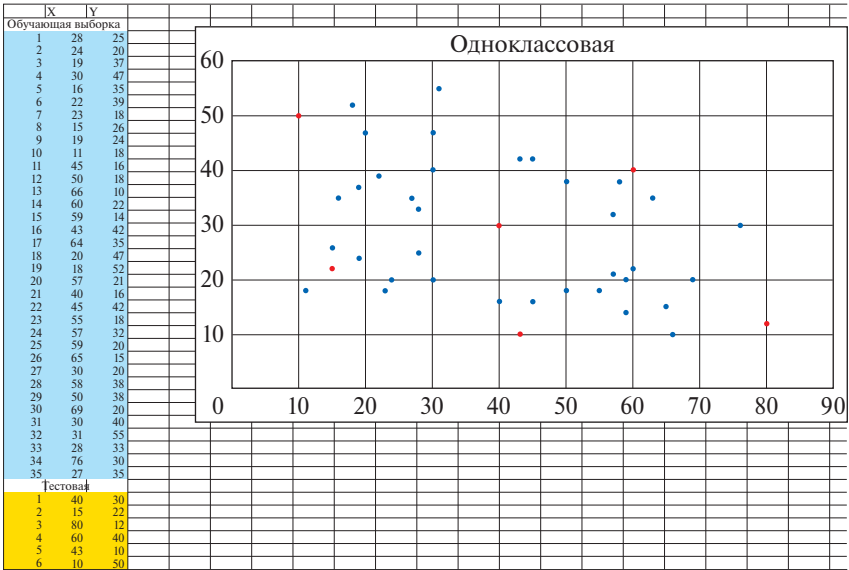
- Медицинская диагностика (Medical Diagnosis)

- Сейсмология

Рассмотрение начнем с простого примера. Пусть есть следующие обучающая и тестовая выборки. Необходимо определить, какие объекты тестовой выборки относятся к данному классу.

Продолжаем рассуждать, а какие возможны варианты решающих правил? Первое, что приходит на ум, — это воспользоваться методом ближайших соседей. Наиболее часто именно так и поступают. Основным недостатком метода ближайших соседей является то, что никакого обучения не происходит, так как необходимо хранить всю обучающую выборку. Поэтому мы рассмотрим другое решающее правило, которое, на наш взгляд, лучше подходит для решения и понимания одноклассовых задач. Принципиально суть алгоритма состоит в следующем:

1. Некоторым образом оконтуриваем обучающее множество



2. Если тестовая точка внутри контура, относим ее к данному множеству, если нет, то

3. Определяем степень близости к контуру:

Если точка расположена к контуру ближе порогового значения, относим ее к данному классу.

Если точка расположена к контуру дальше порогового значения, не относим ее к данному классу.

Возникает вопрос: «Какими гиперповерхностями можно сделать оконтуривание?» Конечно, это может быть гиперсфера или гиперэллипсоид, и мы покажем, как это делается, но чуть погодя, так как хотим показать вам способ, который по причине вычислительных сложностей использовался крайне редко. В настоящее время имеются как способы решения, так и вычислительные мощности, но данный подход так и остался недооцененным. Это выпуклая оболочка.

4.2. ВЫПУКЛЫЕ ОБОЛОЧКИ

Для того чтобы двигаться дальше, нам необходимо привести определения нескольких базовых математических понятий.

Выпуклым будем называть множество, в котором все точки отрезка, проведенного между его любыми двумя точками, также принадлежат данному множеству.

«Выпуклой оболочкой (ВО) множества X называется наименьшее выпуклое множество, содержащее X , то есть такое выпуклое множество, содержащее данную фигуру, что оно содержится в любом другом выпуклом множестве, содержащем данную фигуру».

<https://ru.wikipedia.org/wiki>

Выпуклая оболочка множества X обычно обозначается $\text{Conv } X$

Выпуклая линейная комбинация (ВЛК) — это линейная комбинация точек, в которой все коэффициенты неотрицательны и их сумма равна единице.

Существует достаточно большое количество алгоритмов построения ВО. К сожалению, многие алгоритмы (Грэхэма, Джарвиса, Эндрю, Чена и др.) позволяют находить ВО только для случая двумерного пространства. Для реальных задач такие алгоритмы не подходят.

Как мы уже знаем, алгоритмы решения ЗЛП являются высокоэффективными и позволяют решать задачи большой размерности в многомерных пространствах. Поэтому достаточно логичным было бы свести задачу построения ВО к задаче ЗЛП. В 1994 г. Р.М. Pardalos, Y. Li, W.W. Hager предложили именно такой подход [74]. Они исходили из того, что если точка множества является внутренней, то она может быть представлена как выпуклая линейная комбинация других точек множества. Если же точка является вершиной многогранника (в их терминологии экстремальная точка), то она не может быть представлена как ВЛК других точек, т.е. ее можно представить только через саму себя. Приведем модель в том виде, в каком она была предложена авторами:

$$\text{LP1: } \text{s.t. } \sum_{i \in I}^{\min x_j} x_i a_i = a_j, \quad \sum_{i \in I} x_i = 1, \quad x_i \geq 0 \forall i \in I,$$

где I — множество всех точек;

j — индекс проверяемой на экстремальность точки;

a_i — параметры i -й точки множества I ;

x_i — коэффициенты ВЛК.

Данный подход позволяет для каждой точки множества по отдельности определить, является она экстремальной (крайней) или нет. Последовательное решение ЗЛП в виде, предлагаемом Пардалосом к каждой точке множества по отдельности, позволит определить множество экстремальных точек. Все остальные точки исходного множества являются внутренними, т.е. могут быть представлены как ВЛК крайних точек.

Для решения задач методом Пардалоса понадобятся следующие клише:

```
/* Разложение вектора по базису остальных */
int n = ...; // число параметров
range i = 1..n; // индекс параметра
int m = ...; // число точек базиса
range j = 1..m; // индекс точки
float P[i][j] = ...; // параметры точек базиса
float R[i] = ...; // параметры вектора разлагаемого по базису
/* искомые переменные */
dvar float a[j]; // коэффициенты разложения
subject to {
forall(i in i)
sum(j in j) P[i][j]*a[j]==R[i];
};

/* Простая проверка точек на ВЛК */
int n = ...; // число параметров
range i = 1..n; // индекс параметра
int m = ...; // число точек базиса
range j = 1..m; // индекс точки
float P[i][j] = ...; // параметры точек базиса
float R[i] = ...; // параметры вектора разлагаемого по базису
/* искомые переменные */
dvar float +a[j]; // коэффициенты разложения!! ВНИМАНИЕ
должны быть положительными
subject to {
forall(i in i)
sum(j in j) P[i][j]*a[j]==R[i];
sum(j in j) a[j]==1;/// ВНИМАНИЕ сумма=1

forall(j in j) a[j]>=0; // ВНИМАНИЕ требуем жесткого выполнения
условий неотрицательности коэффициентов
};

/* ПОСЛЕДОВАТЕЛЬНАЯ ПРОВЕРКА ТОЧЕК МНОЖЕСТВА
НА ЭКСТРЕМАЛЬНОСТЬ */
int n = ...; // число параметров
range i = 1..n; // индекс параметра
int m = ...; // число точек
range j = 1..m; // индекс точки
float P[j][i] = ...; // параметры точек множества
```

```

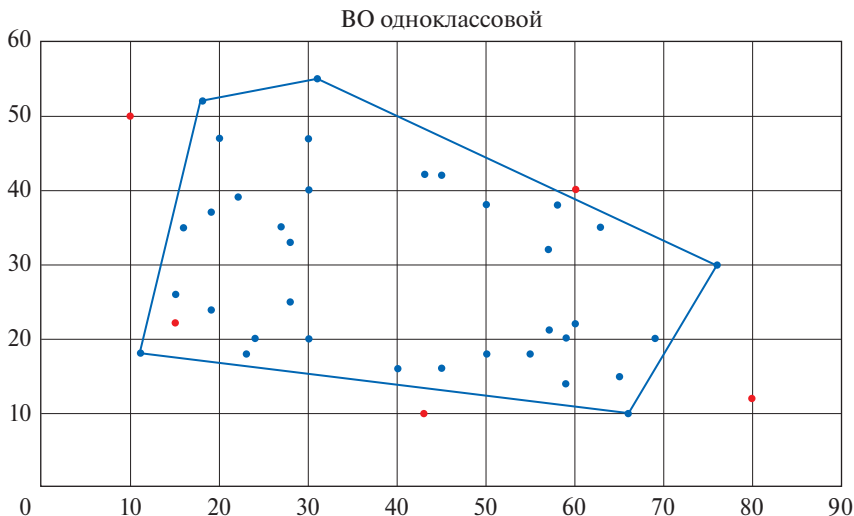
/* искомые переменные */
dvar float+a[j]; // коэффициенты точек в выпуклой линейной
комбинации
int t = 1; // это для первой точки, если запустить цикл то для всех
minimize a[t];
subject to {
forall(i in i)
sum(j in j) P[j][i]*a[j]==P[t][i];
sum(j in j) a[j]==1;
};

```

Если рассматриваемых точек немного, то последовательно изменяя t от 1 до m , мы проверим все точки на экстремальность. Если надо проверить на экстремальность большое число точек, то каким-то образом надо организовать цикл. Это достаточно просто сделать, если вызывать оптимизатор (не обязательно CPLEX) из любого языка программирования, где такая возможность предусмотрена. Именно так и следует делать, но нам не хочется ломать логику изложения и перескакивать с одной темы на другую.

4.3. РЕШЕНИЕ ОДНОКЛАССОВЫХ ЗАДАЧ ВЫПУКЛЫМИ ОБОЛОЧКАМИ

Теперь, когда мы уже умеем строить выпуклые оболочки множеств, вернемся к идее оконтуривания обучающего множества в одноклассовых задачах. Рассмотрим условный пример, приведенный на следующем рисунке.



Можно сформулировать простое решающее правило для одно-классовых задач: если тестовая точка находится внутри ВО, то она принадлежит данному классу, если — за пределами ВО, то не принадлежит.

Конечно, возможны различные варианты решающих правил. Их можно строить на оценке близости тестируемой точки к ВО. Для этого в рамках задачи квадратичного программирования надо найти евклидово расстояние от тестируемой точки до ВО. В дальнейшем мы покажем, как это делается, а пока рассмотрим оконтуривание множества другими гиперповерхностями.

Мы сознательно отложили рассмотрение этого вопроса до сего момента. Конечно, мы могли и ранее оконтурить все точки обучающей выборки окружностью или эллипсом (гиперсферой, гиперэллипсоидом), но: Зачем работать со всеми точками, если все точки кроме экстремальных лежат внутри ВО? Поэтому далее можем работать только с экстремальными точками.

У читателя может возникнуть вопрос: Если мы построили ВО, то зачем еще искать другие контуры? Ответ будет следующим: Такая потребность может возникнуть при решении практических задач. Условный пример мы приводим на плоскости. Поэтому форму множества можем оценить визуально. При работе с многомерными пространствами такая возможность отсутствует, но в некоторых случаях хочется понять, к какой геометрической фигуре ближе обучающее множество. Более того, такие навыки могут понадобиться при поиске эталонных решений, чему будет посвящена отдельная глава.

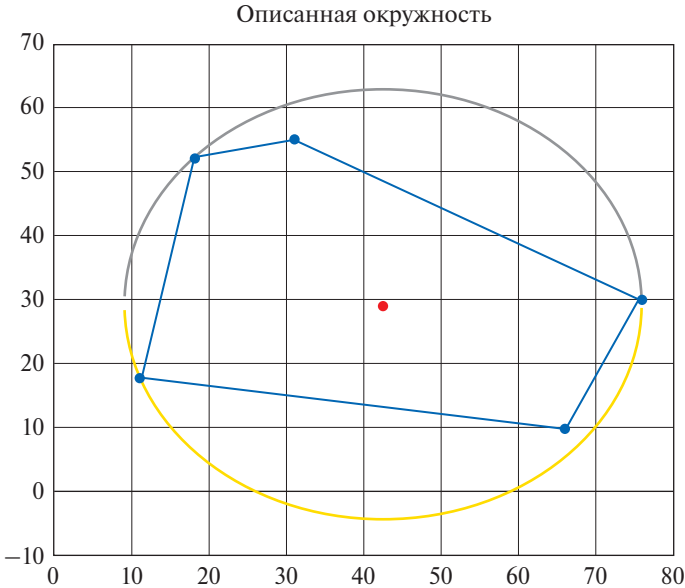
Для того чтобы оконтурить сферой, вам понадобится следующее клише:

```
int n = ...; // число параметров
range i = 1..n; // индекс параметра
int m = ...; // число экстремальных точек
range j = 1..m; // индекс экстремальной точки
float E[j][i] = ...; // параметры точек множества
/* искомые переменные */
dvar float y[i]; // координаты центра
dvar float r; // квадрат минимаксного расстояния от экстремальных точек
minimize r;
subject to {
forall(j in j) sum(i in i) (E[j][i] - y[i])^2 <= r;
};
```

У вас уже достаточно опыта, чтобы понять данное клише. Единственное новое для вас — это то, что система ограничений нелинейна, но это задача квадратичного программирования, и она тоже успешно решается в IBM ILOG CPLEX.

Решение будет выглядеть следующим образом:

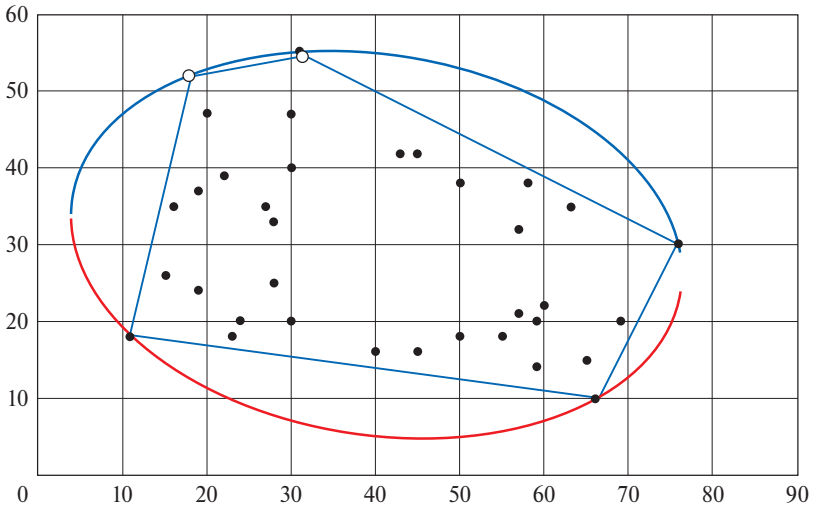
		x	y		
		/*экстремальные*/			r
1	$E=[[$	11	18	$],$	33,47409
2	$[$	66	10	$],$	30,33704
3	$[$	18	52	$],$	33,47409
4	$[$	31	55	$],$	28,23546
5	$[$	76	30	$]];$	33,47409
	Координаты центра сферы			r^2	r
		42,53481	29,22813	1120,5	33,47387



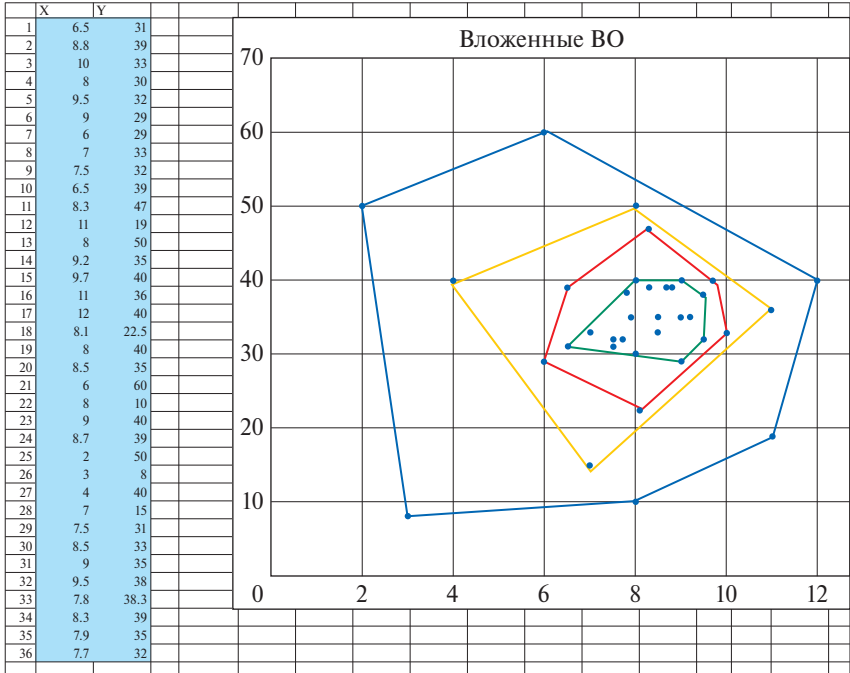
Для построения эллипса уже есть все необходимые знания. Вам просто необходимо вспомнить, что эллипс — это кривая второго порядка, найти формулу кривой второго порядка, перейти в новое пространство признаков и решить эту задачу самостоятельно.

Графически это будет выглядеть следующим образом:

Эллипс, описанный вокруг ВО



При решении одноклассовых задач методом выпуклых оболочек в ряде случаев бывает полезно построить многоуровневую систему ВО. Не вдаваясь в подробности, поясним это на условном примере.



Можно сказать, что приведенный рисунок напоминает некоторый набор матрешек.

Данный прием может быть полезен, когда вы хотите не просто дать ответ: Да или Нет, а хотите дать оценку вероятности правильного отнесения к данному классу. Очевидно, что если тестируемый объект принадлежит самой внутренней ВО, то вероятность отнесения его к данному классу должна быть выше, чем если бы он находился между первой и второй ВО.

4.4. РЕШЕНИЕ МНОГОКЛАССОВЫХ ЗАДАЧ ВЫПУКЛЫМИ ОБОЛОЧКАМИ

Рассмотрим возможность применения ВО в многоклассовых задачах. Допустим, мы имеем многоклассовую задачу. Тогда:

1. Для любого множества обучающей выборки, используя модели МП, можно определить экстремальные точки.

2. Зная экстремальные точки множеств, все точки конкретного множества можно представить как ВЛК экстремальных.

3. Если ни одну из точек одного множества нельзя представить как ВЛК экстремальных точек другого, то ВО множеств не пересекаются, т.е. множества линейно делимы.

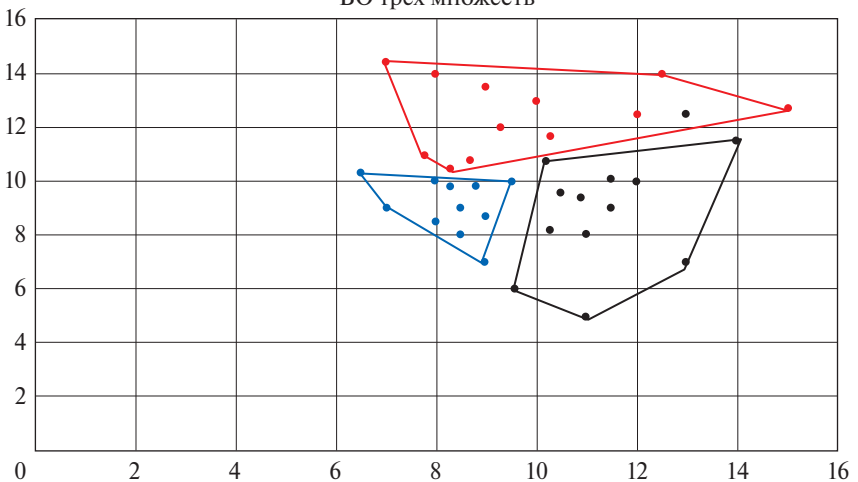
4. Если ВО множеств пересекаются, но в одной из ВО объекты только одного класса, то множества линейно не делимы, но делимы выпуклыми оболочками. Назовем такие множества выпукло-отделимыми (ВОМ).

Приведем графическую интерпретацию сказанного. Пусть имеется три множества:

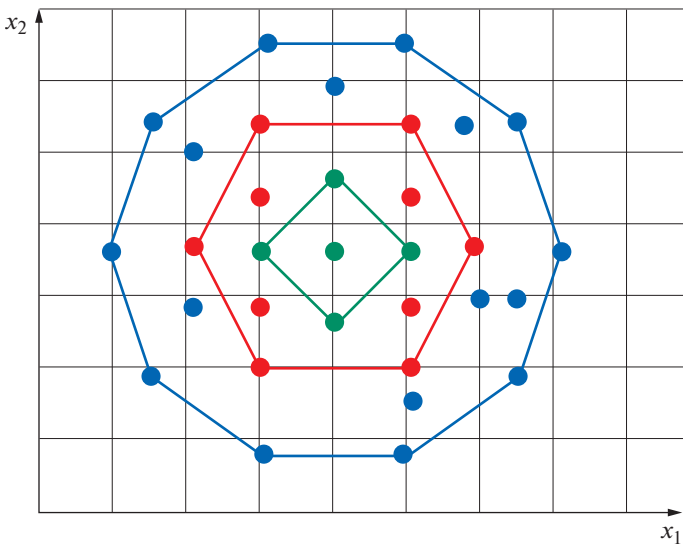
		x	y		экстремальные
1	P=[8,8	9,8],	
2	[6,5	10,3],	1
3	[9	7],	1
4	[9,5	10],	1
5	[8	10],	
6	[9	8,7],	
7	[7	9],	1
8	[8	8,5],	
9	[8,5	8],	
10	[8,5	9],	
11	[8,3	9,8]];	
1	P=[10,3	11,7],	
2	[7,8	11],	1
3	[15	12,7],	1

		x	y	экстремальные	
4	[7	14,4]	1
5	[13	12,5]	
6	[12	12,5]	
7	[8,3	10,5]	1
8	[8,7	10,8]	
9	[12,5	14]	1
10	[8	14]	
11	[10	13]	
12	[9	13,5]	
13	[9,3	12]];	
1	P=[[10,5	9,6]	
2	[10,2	10,8]	1
3	[10,3	8,2]	
4	[11	5]	1
5	[14	11,5]	1
6	[12	10]	
7	[11,5	10,1]	
8	[10,9	9,4]	
9	[11,5	9]	
10	[13	7]	1
11	[11	8]	
12	[9,6	6]];	1

ВО трех множеств



ВО всех множеств взаимно не пересекаются. Значит, все множества попарно линейно разделимы.



ВО всех множеств взаимно пересекаются, но все множества BOM.

Посмотрим, что получим, если данный подход применим к задаче Ирисы. Изначально имеем три обучающих множества:

Setosa, Versicolor и Virginica. Каждое множество изначально содержит по 50 объектов при четырех параметрах. Самостоятельно выполните следующую последовательность действий:

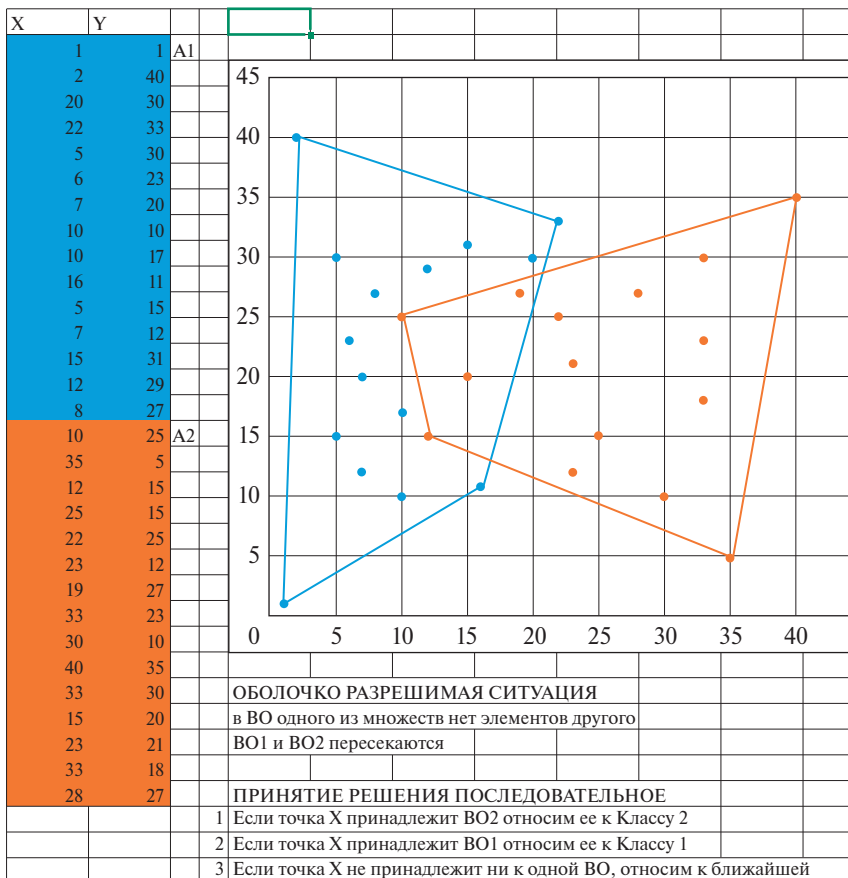
1. Средствами Excel избавьтесь от дублей в каждом множестве.
2. Найдите экстремальные точки каждого множества, используя клише.

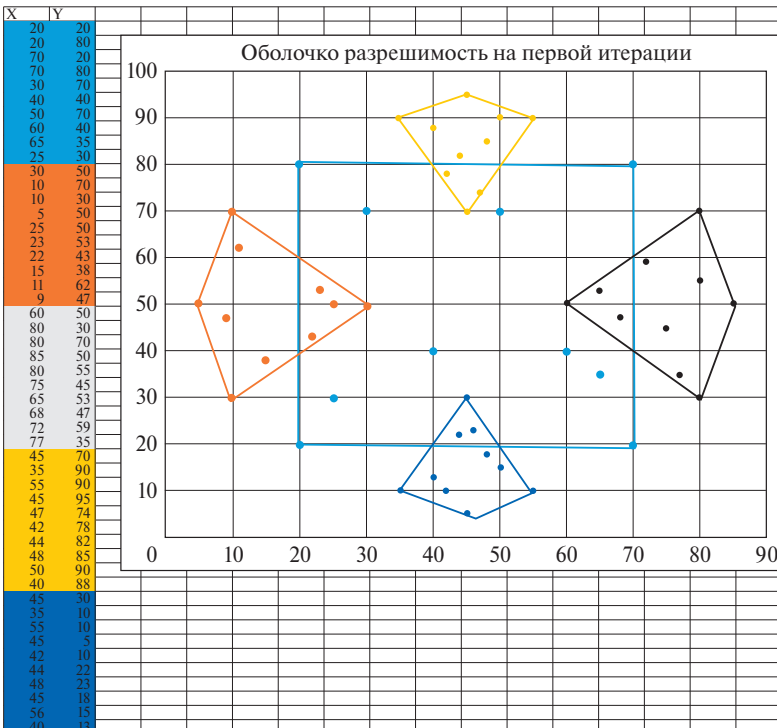
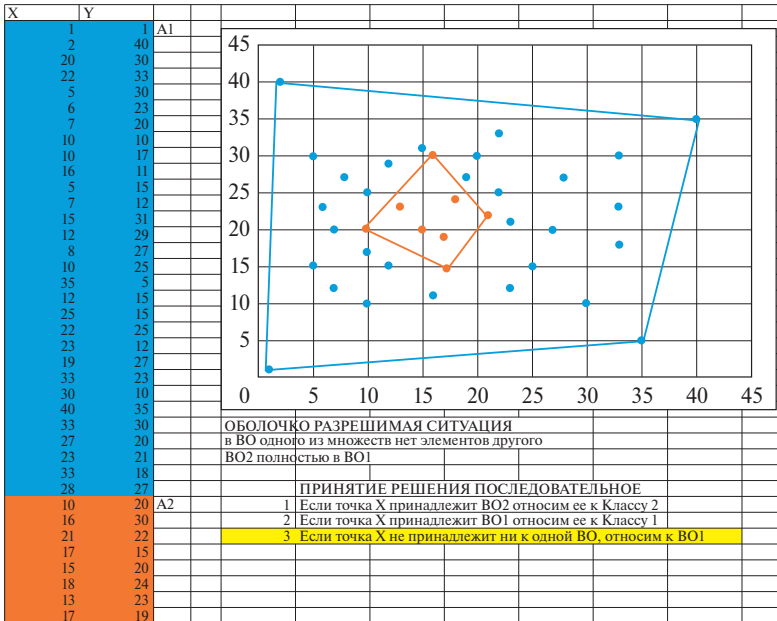
С учетом того что в множестве Setosa имеется три дублирующих друг друга объекта, в дальнейшем рассматриваем 48 объектов. Аналогично, в Virginice два дубля, поэтому рассматриваем 49 объектов. В Setosa экстремальных точек 23, в Versicolor и Virginica – по 26 в каждом.

Проверяем все объекты каждого множества на принадлежность ВО других множеств. Естественно, что Setosa является BOM, и поэтому его можно исключить из дальнейшего рассмотрения. Однако и Versicolor тоже BOM, так как ни один из объектов Virginica не принадлежит ВО Versicolor. Значит, и его можно исключить из дальнейшего рассмотрения. Так как в такой ситуации остается только одно множество Virginica, то ситуация в целом является оболочко разрешимой.

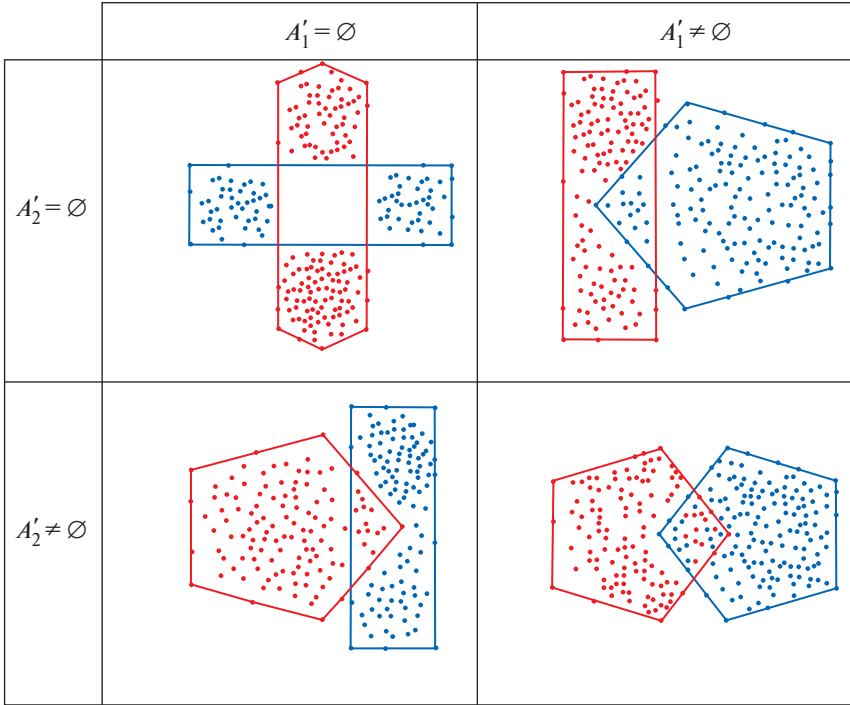
Отметим, что при использовании предлагаемого подхода классификация на обучающих множествах будет полностью совпадать с классификацией Фишера, а объекты тестовых множеств будут классифицироваться однозначно.

Приведем еще примеры оболочко разрешимых ситуаций:





Если рассматривать двухклассовые задачи, то возможны четыре варианта пересечения их ВО.



Проиллюстрируем метод выпуклых оболочек (МВО) на простом примере. Пусть имеются три обучающих множества:

Шаг 1. Находим экстремальные точки каждого множества (строим ВО).

		x	y	
1	P=	9,8	9,8]
2	[6,5	11]
3	[9	7]
4	[9,5	10]
5	[8,5	12]
6	[9,2	11,5]
7	[7	9]
8	[8	8,5]
9	[8,5	8]
10	[8,5	9]
11	[8,5	11,5]

Шаг 2. Определяем для каждой точки каждого множества, в ВО каких множеств она попадает (т.е. определяем, какие ВО взаимно пересекаются).

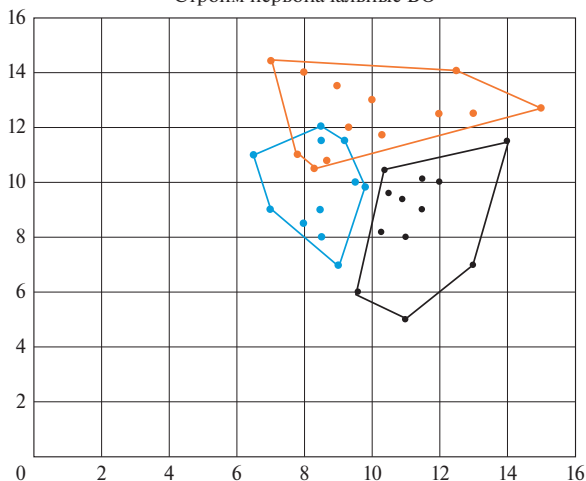
	Обучающие множества	Экстремальная		Другая ВО
		x	y	
1	P=	9,8	9,8]
2	[6,5	11]
3	[9	7]
4	[9,5	10]
5	[8,5	12]
6	[9,2	11,5]
7	[7	9]
8	[8	8,5]
9	[8,5	8]

		x	y	
1	P=[[10,3	11,7]];
2	[7,8	11],
3	[15	12,7],
4	[7	14,4],
5	[13	12,5],
6	[12	12,5],
7	[8,3	10,5],
8	[8,7	10,8],
9	[12,5	14],
10	[8	14],
11	[10	13],
12	[9	13,5],
13	[9,3	12],
1	P=[[10,5	9,6]];
2	[10,4	10,4],
3	[10,3	8,2],
4	[11	5],
5	[14	11,5],
6	[12	10],
7	[11,5	10,1],
8	[10,9	9,4],
9	[11,5	9],
10	[13	7],
11	[11	8],
12	[9,6	6]];

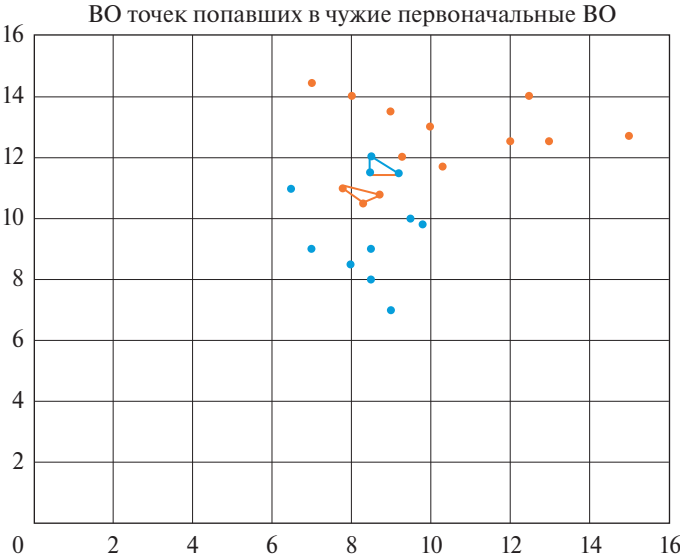
	Обучающие множества	Экстремальная	Другая ВО	
	x	y		
10	[8,5	9],
11	[8,5	11,5],
1	P=[[10,3	11,7]];
2	[7,8	11],
3	[15	12,7],
4	[7	14,4],
5	[13	12,5],
6	[12	12,5],
7	[8,3	10,5],
8	[8,7	10,8],
9	[12,5	14],
10	[8	14],
11	[10	13],
12	[9	13,5],
13	[9,3	12],
1	P=[[10,5	9,6]];
2	[10,4	10,4],
3	[10,3	8,2],
4	[11	5],
5	[14	11,5],
6	[12	10],
7	[11,5	10,1],
8	[10,9	9,4],
9	[11,5	9],
10	[13	7],
11	[11	8],
12	[9,6	6]];

Шаг 3. Строим ВО для точек, попавших в не свою первоначальную ВО.

Строим первоначальные ВО



Шаг 4. Проверяем вновь полученные ВО на взаимное пересечение. Так как взаимопересечения нет, завершаем процесс построения решающего правила.



Конечно, изложение здесь схематичное и не совсем строгое. Более полное изложение с доказательством соответствующих теорем мы приводим в [58].

Продолжим изложение метода. Пусть есть следующее контрольное множество, которое необходимо разметить:

		тестовые		
		x	y	
1	T=[[14	8];
2	T=[[14	14];
3	T=[[6	12];
4	T=[[9	11];
5	T=[[8	10];
6	T=[[10	12];
7	T=[[12	8];

Начнем разметку:

Шаг 1. Определяем, к каким ВО первого уровня относится каждая тестируемая точка.

		тестовые				
		х	у			
1	T=[14	8];	вне	
2	T=[14	14];	вне	
3	T=[6	12];	вне	
4	T=[9	11];	синяя	красная
5	T=[8	10];	синяя	
6	T=[10	12];	красная	
7	T=[12	8];	серая	

Точки 5–7 классифицируются однозначно.

Шаг 2. Точки 1–3 находятся вне всех ВО, поэтому необходимо определить, к какой ВО первого уровня каждая из них ближе. То есть находим расстояние от каждой из них до всех ВО и классифицируем по ближайшему расстоянию. Для определения кратчайшего расстояния от точки до ВО необходимо решить следующую задачу квадратичного программирования:

$$\min \sum_{i \in I} (T_i - v_i)^2, \quad (11.1)$$

$$\sum_{j \in J} a_j = 1, \quad (11.2)$$

$$\sum_{j \in J} C_{ij} * a_j = v_i \quad i \in I, \quad (11.3)$$

$$a_j \geq 0, \quad (11.4)$$

где I – множество параметров (размерность пространства);

J – множество экстремальных точек;

i, j – индексы соответствующих множеств;

C_{ji} – i -я координата j -й экстремальной точки;

T_i – координаты тестовой точки;

v_i – i -я координата точки ВО, ближайшей к тестовой точке;

a_j – коэффициенты ВЛК.

В кодах IBM ILOG CPLEX это будет выглядеть следующим образом:

```
int n = ...; // число параметров
range i = 1..n; // индекс параметра
int m = ...; // число экстремальных точек
range j = 1..m; // индекс точки
```

```

float C[j][i]=...; // координаты (параметры) экстремальных точек
float T[i]=...; // координаты тестовой точки, от которой ищем
минимальное расстояние до ВО
/* искомые переменные*/
dvar float+ a[j]; // коэффициенты ВЛК
dvar float v[i]; // координаты точки ВО ближайшей к тестовой
minimize sum(i in i) (T[i]- v[i])^2;
subject to {
sum(j in j) a[j]==1; // условия ВЛК
forall (i in i) sum (j in j) C[j][i]*a[j]==v[i]; // вычисление координат
// ближайшей точки, можно перенести в целевую, результат
будет тот же, // но значение координат надо будет вычислять
отдельно.
};

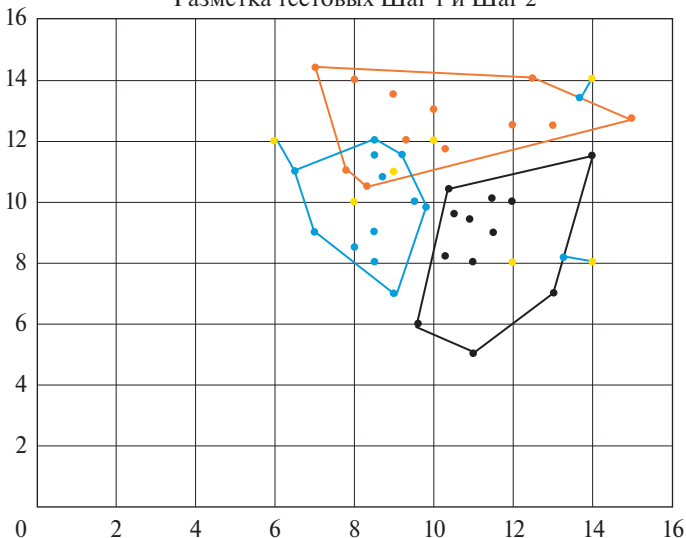
```

Так, вначале находим ближайшую точку, а затем вычисляем евклидово расстояние до нее. Можно, конечно, вычислять евклидово расстояние сразу в модели, но это переусложнит модель.

Получаем следующий результат:

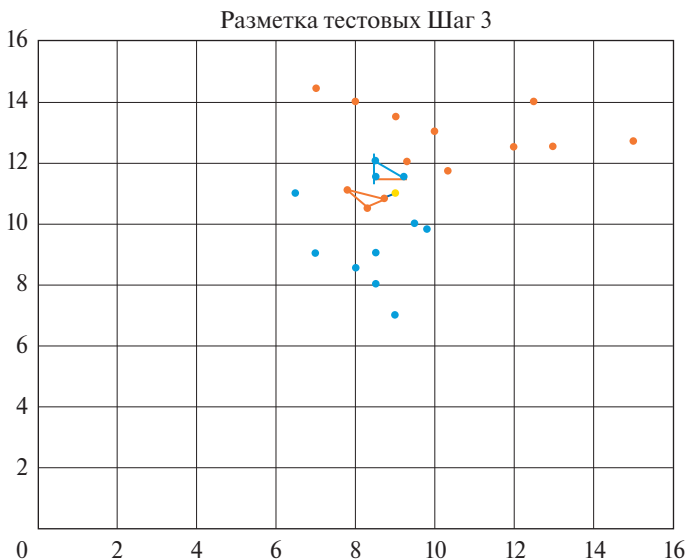
		тестовые			ближайшие к тестовым			расстояние
		x	y		x	y		
1	T=[14	8];	серая	13,259	8,1647	0,759083
2	T=[14	14];	красная	13,681	13,386	0,691923
3	T=[6	12];	синяя	6,5	11	0,5

Разметка тестовых Шаг 1 и Шаг 2



Шаг 3. Вычисляем кратчайшее расстояние от точки 4 до синей и красной ВО второго уровня. Получаем:

	тестовые				ближайшие к тестовым		расстояние	
	x	y			x	y		
T=[9	11];	красная	8,7	10,8		0,360555
				синяя	9	11,5		0,5



То есть точку 4 классифицируем как красная.

Примечание: На третьем шаге мы сразу начали вычислять расстояние до конкретных ВО второго уровня. Если бы точка 4 принадлежала одной из них, то расстояние до нее равнялось бы нулю.

Таким образом, все точки классифицированы. Достоинством МВО является его однозначность на этапе построения решающего правила (если нет абсолютно одинаковых объектов в разных классах обучающей выборки, что, наверное, надо считать за ошибку в исходных данных). На этапе разметки контрольной выборки неоднозначность может возникнуть, только если тестируемая точка не принадлежит ни одной ВО и находится на совершенно одинаковом расстоянии от нескольких. Аналогичная проблема может возникнуть и в методе ближайших соседей, даже если число их равняется некоторому «к». Вполне может найтись точка, которая находится на одинаковом расстоянии от «к» объектов как одного класса, так и другого. В таком случае, видимо, надо честно сказать, что ситуация действительно неоднозначная.

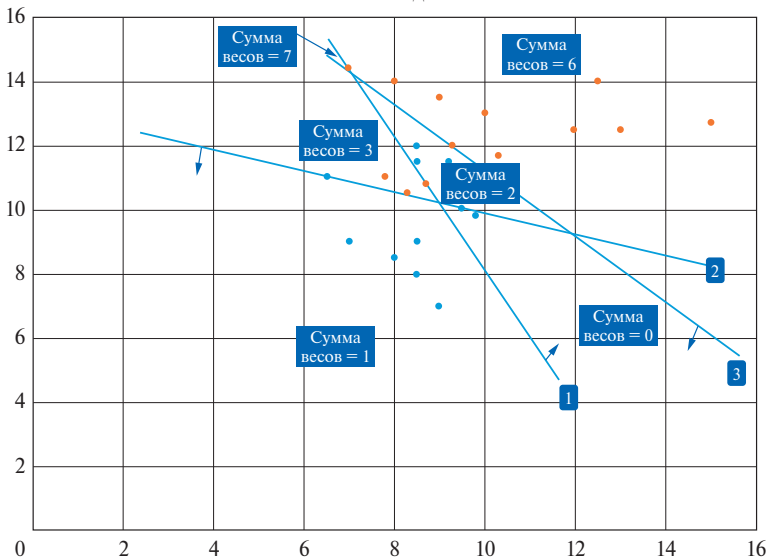
4.5. ВЗАИМОСВЯЗЬ МЕТОДА КОМИТЕТОВ И МВО, ЭТАЛОННЫЕ СОСТОЯНИЯ

Попробуем разграничить синее и красное множества предыдущего примера методом комитетов, а именно построим комитет старшинства как наиболее универсальный. Например, возможен следующий КС:

	комитет						веса			m=	2
	x	y	b				1	2	4		
	-2,11465	-1	29,2335								
	0,33333	1	-13,177								
	1,04348	1	-21,694								
	проверка										
1	9,8	9,8		-1,32	-0,11	-1,67	0	0	0	0	
2	6,5	11		4,469	-0,01	-3,91	1	0	0	1	
3	9	7		3,175	-3,18	-5,3	1	0	0	1	
4	9,5	10		-0,88	-0,01	-1,78	0	0	0	0	
5	8,5	12		-0,77	1,657	-0,82	0	1	0	2	
6	9,2	11,5		-1,75	1,39	-0,59	0	1	0	2	
7	7	9		5,41	-1,84	-5,39	1	0	0	1	
8	8	8,5		3,792	-2,01	-4,85	1	0	0	1	
9	8,5	8		3,234	-2,34	-4,82	1	0	0	1	
10	8,5	9		2,234	-1,34	-3,82	1	0	0	1	
11	8,5	11,5		-0,27	1,157	-1,32	0	1	0	2	
1	10,3	11,7		-4,28	1,957	0,753	0	1	1	6	
2	7,8	11		1,716	0,423	-2,56	1	1	0	3	
3	15	12,7		-15,2	4,523	6,658	0	1	1	6	
4	7	14,4		0,01	3,557	0,01	1	1	1	7	
5	13	12,5		-10,8	3,657	4,371	0	1	1	6	
6	12	12,5		-8,68	3,323	3,327	0	1	1	6	
7	8,3	10,5		1,157	0,09	-2,53	1	1	0	3	
8	8,7	10,8		0,01	0,523	-1,82	1	1	0	3	
9	12,5	14		-11,2	4,99	5,349	0	1	1	6	
10	8	14		-1,71	3,49	0,653	0	1	1	6	
11	10	13		-4,94	3,157	1,74	0	1	1	6	
12	9	13,5		-3,33	3,323	1,197	0	1	1	6	
13	9,3	12		-2,46	1,923	0,01	0	1	1	6	

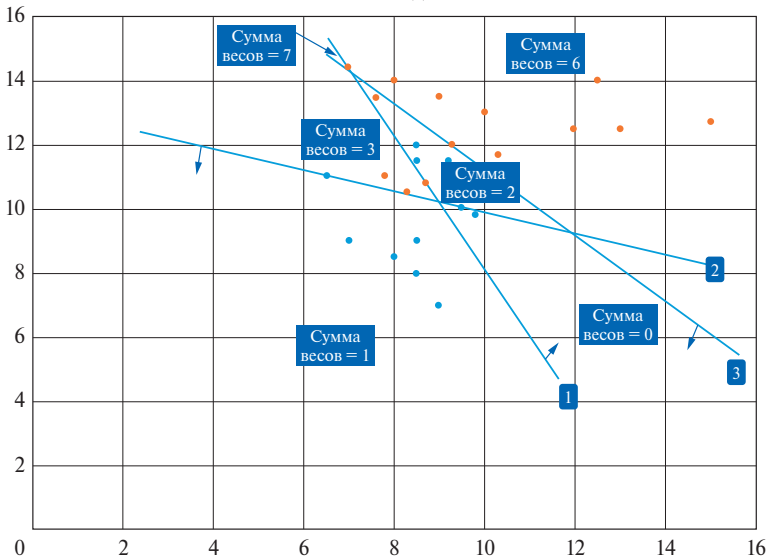
Графически это выглядит следующим образом:

КС на всех данных



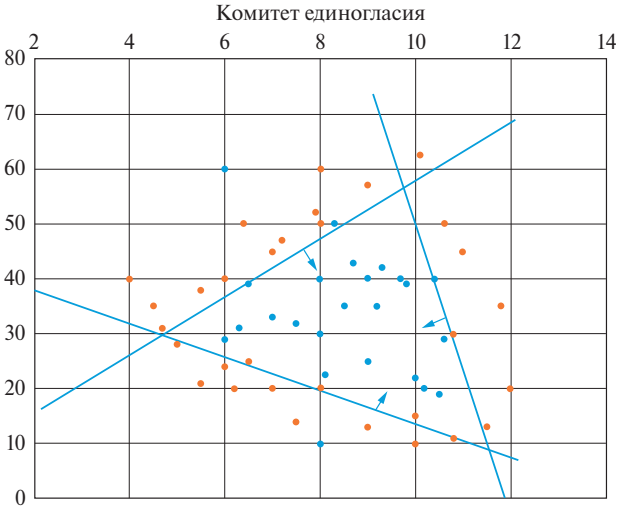
По сути, если объединить секторы с весом, равным 1, и весом, равным 0, и во всех секторах построить ВО, то мы получим решение, совпадающее с решением МВО. Теперь допустим, что есть тестируемая точка с координатами (7.6, 13.5). Желтая точка в секторе – сумма весов = 2.

КС на всех данных

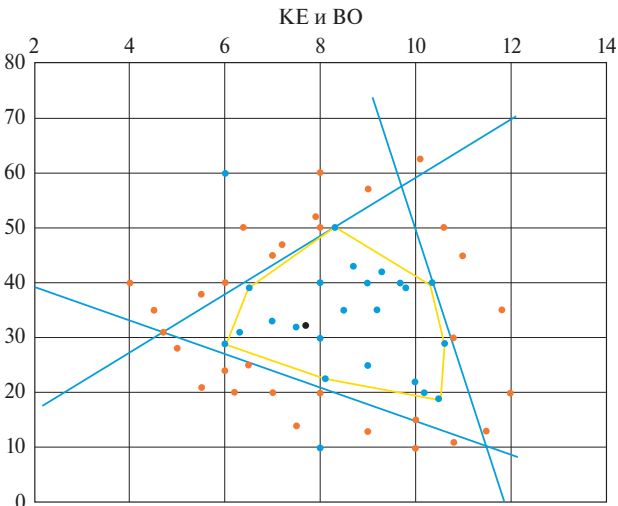


Метод комитетов отнесет ее к синим, а МВО — к красным. В данной ситуации, исходя из близости объектов разных классов, эту тестовую точку действительно следует отнести к классу красных.

Рассмотрим другой пример. Графически это выглядит следующим образом:



Такие задачи достаточно часто встречаются при отборе ответственных деталей методами неразрушающего контроля или при подборе оптимальных параметров сложных, дорогостоящих или опасных производственных процессов. В этом случае от вас, скорее всего, потребуют максимально жесткого решающего правила.



Одним из способов ужесточения решающего правила может быть построение ВО внутри КЕ.

Согласитесь, что решающее правило, ограниченное желтыми линиями, будет более надежным, чем ограниченное синими линиями. Конечно, если использовать его, то при отборе деталей отбраковка будет больше. Однако, если речь идет об отборе деталей для космического аппарата, то, наверное, лучше воспользоваться им, чем решать проблемы, когда аппарат уже в космосе.

На практике Вам могут задать и такой вопрос: А какие параметры, на ваш взгляд, самые надежные (идеальные), чтобы продукция была хорошего качества (процесс был стабилен и т.п.)? Тогда вы естественным образом придете к необходимости найти точку, находящуюся внутри ВО или КЕ, но максимально удаленную от границ. Назовем ее эталонной.

Конечно, это уже задача из области Исследования операций, но ее надо уметь решать, и нам опять поможет Математическое программирование.

Рассмотрим нахождение эталона для КЕ. В данном случае мы имеем выпуклую область, ограниченную системой ограничений, и формулы этих ограничений нам известны. Для того чтобы сформулировать данную задачу как ЗМП, нам необходимо вспомнить формулу расстояния от точки до плоскости.

Пусть задана точка $M_0(x_0, y_0, z_0)$ и плоскость P своим уравнением $Ax + By + Cz + D = 0$. Расстояние от точки M_0 до плоскости находят по формуле:

$$d = \frac{|Ax_0 + By_0 + Cz_0 + D|}{\sqrt{A^2 + B^2 + C^2}}$$

Тогда ЗМП выглядит следующим образом:

$$\begin{aligned} \sum_{i \in I} a_i^t * x_i + b^t &= \sqrt[2]{\sum_{i \in I} (a_i^t)^2} * r^t \quad t \in T \\ r^t &\geq \varphi \quad t \in T \\ \max \varphi & \end{aligned}$$

где x_i – искомые координаты точки эталона;

a_i^t – коэффициент гиперплоскости t для i -го параметра;

b^t – свободные члены гиперплоскостей;

a_i^t и b^t – найдены при построении КЕ;

r^t – длина перпендикуляра от эталонной точки до t -й гиперплоскости;

φ – максимальная длина перпендикуляра.

Приведем модель в кодах CPLEX:

```

/* центр системы неравенств (эталонная точка) */
int n = ...; // размерность пространства (число параметров наблюдений)
range i = 1..n; // индекс параметра
int m = ...; // число гиперплоскостей
range j = 1..m; // индекс гиперплоскости
float A[j][i]=...; // матрица коэффициентов гиперплоскостей
float B[j]=...; // свободные члены гиперплоскостей
float Sqrt[j]=...; // корень из суммы квадратов коэф. гиперплоскости

/* в ряде случаев могут понадобиться эти ограничения
float VG[i]=...; // верхняя граница x[i]
float NG[i]=...; // нижняя граница x[i]
*/!!! ограничения на границы могут сделать задачу противоречивой

/* искомые переменные*/
dvar float x[i]; // координаты центра системы (эталон)
dvar float + r[j]; // расстояние от центра до гиперплоскости
dvar float + minr; // минимальное расстояние от центра до гиперплоскости

maximize minr;
subject to {
forall(j in j)
sum(i in i) A[j][i]*x[i]+B[j]==Sqrt[j]*r[j];
forall(j in j) r[j]>=minr;
/*
forall(i in i) x[i] <= VG[i];
forall(i in i) x[i] >= NG[i];
*/
};

```

Приведем схему **одного из возможных** для данного примера КЕ:

			Данные для вычисления эталона			
			a1	a2	свободн член	константы
		комитет →	-25	-1	299	25,01999201
			3	1	-46	3,16227766
			6	-1	1	6,08276253

/*коэффициенты*/						
A=[[-25	-1]			
[3	1]			
[6	-1]];			
/*свободн член*/						
V=[299	-46	1];		
/*константы (корень из суммы квадратов)*/						
			6,08276			
C=[25,01999	3,16227766	3];		
	x1	x2				
эталон ->	7,8927	31,227				
	a1	a2	b			
	-25	-1	299	70,4555	25,01999201	2,816
	3	1	-46	8,9051	3,16227766	2,816
	6	-1	1	17,1292	6,08276253	2,816

Мы не случайно использовали оборот **одного из возможных**. Метод комитетов допускает наличие альтернативных комитетов. Поэтому, используя приведенную выше схему, найдите решение для следующего КЕ:

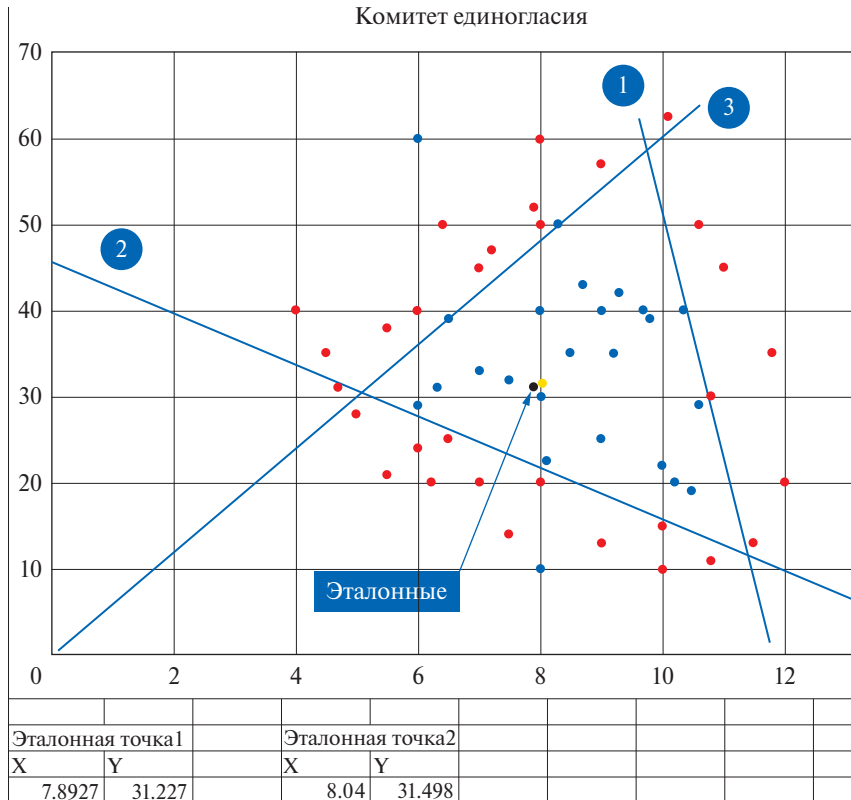
$$3.45x + y - 49.6 \geq e$$

$$6.47222x - y - 2.9694 \geq e$$

$$-150x - y + 1639 \geq e$$

Сравним решения:

Комитет единогласия



Решения, конечно, различаются, но незначительно.

Теперь рассмотрим, как строить эталонные точки для ВО. В этом случае у нас нет гиперплоскостей, есть только экстремальные точки. Конечно, можно попытаться оконтурить ВО гиперплоскостями. Сразу скажем, что для задач большой размерности это теоретически возможно, но очень ресурсоемко. Поэтому в качестве эталона проще взять точку, максимальное расстояние от которой до всех экстремальных точек минимально. Для этого можете воспользоваться следующим клише:

```

int n = ...; // число параметров
range i = 1..n; // индекс параметра
int m = ...; // число экстремальных точек
range j = 1..m; // индекс экстремальной точки
float E[j][i] = ...; // параметры экстремальных точек множества
/* искомые переменные */
dvar float y[i]; // координаты центра
dvar float + x[j]; // коэффициенты ВЛК
    
```

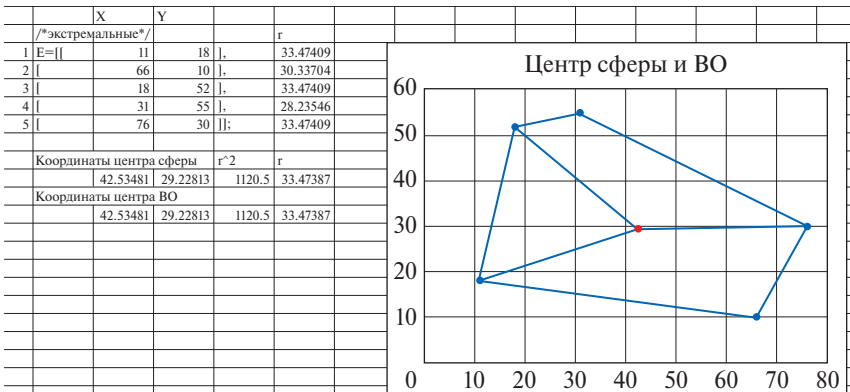


```

dvar float+ r; // минимаксное расстояние от экстремальных точек
minimize r;
subject to {
forall(i in i) sum(j in j) E[j][i]*x[j] == y[i];
sum(j in j) x[j] == 1;
forall(j in j) sum(i in i) (E[j][i] - y[i])^2 <= r;
};

```

Можете попробовать данное клише на примере, который мы рассматривали ранее:



В данном примере координаты центра ВО и центра описанной сферы совпадают.

Теперь рассмотрим случай, когда разделить множества комитетом единогласия невозможно, но можно разделить, например, комитетом старшинства. **Так как каждая область КС, имеющая свой вес, ограничена системой неравенств, то можно использовать те же подходы.** Надо просто правильно записать систему неравенств, в ряде случаев ввести границы на переменные и найти эталоны для каждой области. При этом в каждой области вы можете построить свои ВО и ужесточить решающие правила.

Взаимосвязь метода комитетов и метода выпуклых оболочек может быть использована и по-другому. Если все ВО, необходимые для построения решающего правила, построены, т.е. определены их экстремальные точки, то далее КС можно строить только на основе этих точек, так как остальные могут быть выражены через них, т.е. не быть информативными для построения КС.

Глава 5

ВМЕСТО ПОСЛЕСЛОВИЯ. ФИЛОСОФСКИЕ ПРОБЛЕМЫ МАШИННОГО ОБУЧЕНИЯ

Nihil est in intellectu quod non prius fuerit in sensu...
nisi intellectus ipse.

Нет ничего в интеллекте, чего бы раньше
не было в чувстве... кроме самого интеллекта.

Г. Лейбниц «Новый опыт о человеческом разуме»

Краткий философский взгляд на возможности и условия применения рассматриваемых в монографии цифровых технологий кажется нам необходимым для понимания, эффективного применения и совершенствования этого инструмента. Деятельность программиста, как и деятельность математика, не может быть понята изнутри этой деятельности, вне ее генезиса и связей с другими видами человеческой активности. Философский анализ поясняет, почему программирование на основе математического моделирования кажется авторам монографии более осмысленным, а значит, более адаптивным и интерпретируемым, нежели программирование, заданное правилами и шаблонами языков программирования.

Любой вид человеческой деятельности, вырастая и надстраиваясь над предшествующими, далее приобретает относительную самостоятельность от них, «забывая» о своих корнях. Такая метаморфоза происходит сейчас с ИТ-деятельностью, которая считает себя вполне самодостаточной и способной развиваться «на собственной основе», ограничиваясь усвоением собственного «языка» и правил. Уже создан новый – «цифровой» («он-лайн») – мир, для многих не менее реальный, чем мир «физический» («офф-лайн»). ИТ-специалисты, погруженные (профессионально и ментально) в цифровую реальность, не могут да и не должны рефлексировать по поводу социокультурных и экономико-политических корней и ограничений своей деятельности.

Указанное обстоятельство объясняет, почему предлагаемые в нашей монографии методы машинного обучения сегодня не получили соответствующей их потенциалу разработки и применения. Также это справедливо применительно к ситуации с развитием искусственного интеллекта в целом.

Утвержденная 10 октября 2019 г. Указом Президента РФ № 490 «Национальная стратегия развития искусственного интеллекта на период до 2030 года» (далее – Стратегия) дала «зеленый свет» административной и финансовой поддержке полномасштабной экспансии представителей ИТ-индустрии во все сферы человеческой жизни. Включая ее «мозговой штаб» – процесс принятия решений. Любых.

Искусственный интеллект в Стратегии определяется как «*комплекс технологических решений, позволяющий имитировать когнитивные функции человека (включая самообучение и поиск решений без заранее заданного алгоритма)*».

Соответственно «...*технологии искусственного интеллекта – технологии, основанные на использовании искусственного интеллекта, включая ... интеллектуальную поддержку принятия решений*»¹.

Мы ни в коей мере не отрицаем значимость и благотворность научно-технического прогресса, включая его «цифровую» составляющую, и даже готовы подписаться под словами главного «идейного отца» этой концепции Германа Грефа, что «*искусственный интеллект сегодня – это включатель вывода на новый уровень всех без исключения технологий. Нет ни одной технологии, нет ни одной науки, которая бы сегодня не приобрела принципиально новые возможности в связи с использованием искусственного интеллекта*»².

В то же время мы полностью разделяем отмеченную в Стратегии опасность: «*Отсутствие понимания того, как искусственный интеллект достигает результатов, является одной из причин низкого уровня доверия к современным технологиям искусственного интеллекта и может стать препятствием для их развития...*»

Создание универсального (сильного) искусственного интеллекта, способного, подобно человеку, решать различные задачи, мыслить, взаимодействовать и адаптироваться к изменяющимся условиям, является сложной научно-технической проблемой, решение которой находится на пересечении различных сфер научного знания – естественно-научной, технической и социально-гуманитарной»³.

¹ Цит. по: <https://www.garant.ru/products/ipo/prime/doc/72738946/> выделено нами.

² Цит. по: <http://www.kremlin.ru/events/president/news/60630/> выделено нами.

³ Цит. по: <https://www.garant.ru/products/ipo/prime/doc/72738946/> выделено нами.

Разделяем, но с одной маленькой оговоркой, касающейся постановки социально-гуманитарного знания в перечне «инструментов», необходимых для эффективного решения проблем развития и применения искусственного интеллекта, на последнее — третье — место. В нас говорит не уязвленная профессиональная гордость, а согласие с утверждением немодного ныне мыслителя, что «...кто берется за частные вопросы без предварительного решения общих, тот неминуемо будет на каждом шагу бессознательно для себя “натывать” на эти общие вопросы. А натывать слепо на них в каждом частном случае значит обрекать свою политику на худшие шатания и беспринципность»⁴.

Создание универсального (сильного) искусственного интеллекта — это не только и не столько научно-техническая проблема, сколько общегуманитарная, философская проблема. Еще раз согласимся со Стратегией, что *«Решение этой проблемы может привести не только к позитивным изменениям в ключевых сферах жизнедеятельности, но и к негативным последствиям, вызванным социальными и технологическими изменениями, которые сопутствуют развитию технологий искусственного интеллекта»*⁵. Попробуем разобраться, как философия может поучаствовать в создании универсального (сильного) искусственного интеллекта и, что не менее важно, в заключении некоего общественного договора по его применению.

На первый взгляд, ситуация здесь в чем-то схожа с созданием ядерного оружия, которое, будучи порождением научно-технического прогресса, потребовало определенных гуманитарных и политических усилий по своему ограничению. Хотя точнее было бы сравнить проблемы, возникающие в связи с искусственным интеллектом, с двумя «чумами XX века» (фашизм и коммунизм), которые были вызваны не столько техническими, сколько социальными технологиями. Конечно, превращение последних в эпидемию, охватившую большую часть цивилизованного мира, было бы невозможно без таких «технологических» изобретений, как СМИ (газеты, радио, позднее телевидение). Кстати, антигуманный потенциал «цифровизации» уже тогда был продемонстрирован в ходе создания компанией IBM автоматов для учета всех евреев. За создание подобных устройств глава и основатель IBM Томас Уотсон-старший в 1937 г. был награжден нацистами Орденом Заслуг германского орла⁶.

⁴ Ленин В.И. ПСС. т. 15, с. 368.

⁵ Цит. по: <https://www.garant.ru/products/ipo/prime/doc/72738946/> выделено нами.

⁶ См. IBM and the Holocaust: The Strategic Alliance between Nazi Germany and America's Most Powerful Corporation. New York: Crown Publishers, 200 или https://ru.wikipedia.org/wiki/IBM_%D0%B8_%D0%A5%D0%BE%D0%BB%D0%BE%D0%BA%D0%BE%D1%81%D1%82

Этот экстремальный пример использования «ранних» цифровых технологий «силами зла» демонстрирует нам неожиданный взгляд на связь между вещественным миром (куда мы относим и природу, и социум), подчиняющимся естественным (естественно-историческим)⁷ законам, и «цифровым миром», объекты и законы которого создаются человеком по его воле и желанию и являются «искусственными». Конечно, естественно-историческая реальность (история, социум) и ее законы также являются результатом предшествующей деятельности людей, наделенных волей и сознанием, но результатом объективным (уже данный), т.е. не зависящим от сознания и воли предстоящих здесь и сейчас перед ним людей.

Неожиданность взгляда заключается в том, что явление, именуемое сегодня «искусственным интеллектом», существовало всегда. Точнее, интеллект⁸ как специфическая способность человека «к самообучению и поиску решений без заранее заданного алгоритма» всегда был **не-естественным**. Любой интеллект всегда **искусствен** в том смысле, что он есть результат «выделения» человека из природы (**естественного** состояния, подчиняющегося только физическим/природным законам), реализации им самого себя в ходе созидания над-естественной – культурной – среды обитания.

При этом видимые формы его презентации нам на протяжении человеческой истории были различными. Разделяя критику Е.В. Островским⁹ понятия «искусственный интеллект» (Artificial Intelligence) и неправомочность противопоставления его интеллекту естественному (как не существующему в природе), хотим высказать свою позицию относительно тех «платформ» (носителей), посредством которых интеллектуальная функция человека становится доступной внешнему восприятию. В философии спор о носителях интеллекта (мышления, сознания)¹⁰ часто формулируется

⁷ То есть не зависящим от воли и сознания взаимодействующего с ними человека.

⁸ Интеллект – качество психики, состоящее из способности осознавать новые ситуации, способности к обучению и запоминанию на основе опыта, пониманию и применению абстрактных концепций, и использованию своих знаний для управления окружающей человека средой (перевод статьи из Encyclopaedia Britannica, опубликован в Википедии[4]).

⁹ О некорректности понятия «естественный интеллект» в его противопоставлении «искусственному интеллекту» см. Островский Е.В. «RAI: удаленный интеллект». Цит. по: <https://russiancouncil.ru/analytics-and-comments/analytics/rai-udalyennyu-intellekt/>

¹⁰ Эти термины мы используем как синонимы исключительно в данном контексте противопоставления мыслительной, интеллектуальной, сознательной деятельности человека и деятельности психической, неосознаваемой, бессознательной, которая есть и у животных и может быть зафиксирована экспериментальным путем при анализе нейрофизиологической строения и функционирования нервной системы, микроструктур головного мозга, нейронных связей и т.п.

как вопрос, является ли мышление (интеллект, сознание) функцией мозга как белкового образования или же функцией какого-то иного «субстрата» (носителя).

Поиск ответа на этот вопрос сегодня получил новую актуальность в связи с тем, что изобретение в конце XX в. так называемого искусственного интеллекта создает иллюзию, что эта специфическая человеческая функция **впервые** может быть «вынесена за пределы» человеческого тела (мозга) или даже их «совокупности», и пересажена с «белкового hardware» на «песок», или «кремниевую платформу»¹¹. Под последней понимается не имеющая **естественных** аналогов, т.е. **искусственно** созданная человеком в последние десятилетия, современная компьютерная Сеть – «мыслящая машина», Сверхразум или Искусственный Интеллект. Мы же считаем, что такое «внедрение» Мыслящей Машины как Надличностного Разума в человеческую деятельность и подчинение ею человека произошло не сегодня. Мы разделяем точку зрения, что интеллект **всегда** функционировал в определенной «операционной среде» в качестве различных «инструментальных (=институциональных) приложений»¹² – право, медицина, искусство, политика, экономика и иные социальные и культурные институты.

Поэтому «хромающей» выглядит аналогия между заменой компьютером как «мыслящей машиной» интеллектуальной (мыслительной) функции человека и заменой паровой или электрической машинами мускульной силы человека. Принципиальное различие заключается, во-первых, в том, что мускульная сила, в отличие от мышления (интеллекта), не является специфической характеристикой человека как родового существа. Она присуща и животным, которые, кстати, выполняли до появления машин эту функцию для человека. Во-вторых, «вынесенность» мыслительной (интеллектуальной) функции человека за пределы индивидуально-го тела (и мозга) существовала изначально, вместе с родом человеческим. Точнее, она не могла существовать иначе, нежели «вынесенной» в пространство («операционную среду») **межчеловеческого взаимодействия**, как практического, так и духовного. Индивид вне общества как системы социальных коммуникаций и человеческой цивилизации¹³ – это Маугли – животное, лишённое мышления (интеллекта, сознания)¹⁴.

¹¹ Термин Островского Е.В., см. там же.

¹² Термины Островского Е.В., см. там же.

¹³ О роли культуры и понимании интеллекта не как результата деятельности отдельного мозга, но как функции от тела человеческой цивилизации применительно к анализу дискуссий 60-х годов «о природе» искусственного интеллекта см. А.С. Арсеньев, Э.В. Ильенков, В.В. Давыдов. «Машина и человек, кибернетика и философия» // Ильенков Э.В. Собр. соч. Т. 3. М.: Канон+ РООИ «Реабилитация», 2020. С. 143–161.

¹⁴ См. сноску 10.

Самыми первыми и до сих пор самыми сложными Мыслящими Машинами являются Рынок (экономика) и Государство (политика). Именно они вместе с надстроенными над ними формами общественного сознания и культурными институтами не только обладают способностью «к самообучению и поиску решений без заранее заданного алгоритма», но еще и формируют мышление (интеллект), а заодно и жизнь каждого индивидуума. Чтобы не породить новой терминологии, предлагаем в дальнейшем называть описанные в этом абзаце «социальные мыслящие машины» их традиционным именем – Социальные Институты. Соответственно «мыслящими машинами», к которым относится рассматриваемый в нашей монографии термин «машинное обучение», а также Искусственным Интеллектом мы будем называть созданные с использованием современных цифро-сетевых технологий «комплексы технологических решений, позволяющих имитировать когнитивные функции человека».

При этом будем постоянно помнить, что по своему происхождению и функционалу Социальные Институты есть не что иное как социальные мыслящие машины, т.е. обладающие неконтролируемой, а зачастую и непонятной конкретным индивидам собственной «логикой» (надличностным Интеллектом, Мышлением, Разумом) механизмы взаимодействия этих индивидов.

Социальные Институты изначально не были естественно-природными («белковыми») образованиями, хотя для их функционирования «требовалось» взаимодействие высокоорганизованных «белковых тел» – людей. Но еще больше для них требовалось наличие специальных «носителей», способных выполнять знаковую функцию, т.е. быть не только объектами-самими-по-себе, но и объектами, означающими, несущими информацию (содержание) о чем-то ином, чем они сами не являлись. Такие объекты называются объектами материальной и духовной культуры, и их «культурно-значущее» содержание опредмечивается и распределмечивается («кодируется и раскодируется») в ходе операций с ними каждого нового поколения людей¹⁵. Культурно-знаковые объекты не являются естественно-природными («белковыми»), они искусственно создаются людьми. Поэтому не имеет смысла говорить о цифровых технологиях как переносе Интеллекта «с белка на песок». Его там никогда не было.

Однако революционные события, связанные с интеллектуальной функцией Человека, в последние 50 лет, действительно,

¹⁵ Самой наглядной иллюстрацией функционирования таких Мыслящих Машин является процесс создания письменных текстов и их последующее прочтение, а в последующем – технология массовой книжной печати и массового чтения.

произошли. И они имеют прямое отношение к процессу «переноса» Интеллекта на качественно новый «носитель». Этим новым «носителем» стали так называемые Цифровые Объекты, возникшие в результате развития компьютерных технологий. **Перечислим некоторые основные свойства цифровых объектов, отличающие их от объектов физических (вещественных, материальных)**¹⁶:

1. **Воспроизводимость.** Любой цифровой объект можно бесконечно копировать, не теряя при этом информации.

2. **Неразличимость копий.** Все полученные копии абсолютно идентичны оригиналу.

3. **Отсутствие жесткой связи с материальным носителем.** В отличие от бумажной книги в библиотеке электронная копия книги существует отдельно от устройства, в котором она хранится.

4. **Нелокальность во времени и пространстве.** Цифровые объекты можно практически мгновенно перемещать в пространстве или хранить «в облаке», тем самым полностью избавившись от пространственной принадлежности.

5. **Необходимость использования технических устройств.** Цифровой объект не существует сам по себе, он становится доступным для использования только в результате его взаимодействия с технически сложным объектом.

6. **Динамичность.** Цифровой объект можно непрерывно модифицировать, постоянно изменяя его форму и содержание.

Вышеперечисленные особые свойства цифровых объектов привели к появлению так называемых **цифро-сетевых технологий** (далее – ЦСТ). А ЦСТ произвели настоящую революцию в процессах коммуникации (взаимодействия) между людьми. Произошел качественный скачок в том, что Маркс называл формами общения людей, или общественными отношениями. По сути, возникла возможность новой социальности, имеющей другую связность и скорость распространения идей, схем поведения и общения. В «производство» социальных связей (социальное творчество) стали вовлекаться неизмеримо большее количество ранее пространственно разделенных людей. И как следствие – возможность цифровой перестройки традиционных Социальных Институтов. Так возникли Цифровая экономика, Цифровая (медийная) Политика, Социальные сети как цифровой тип обыденного общения и т.п.

Возникла новая степень свободы в оперировании культурно-знаковыми объектами как посредниками во взаимодействии людей в рамках традиционных Социальных Институтов, что изменило

¹⁶ См. <http://www.unknown.ru/ostraya-tema/6072-infrastruktura-noosfery.html>

саму «ткань» этих Институтов. Одновременно появилась возможность создавать новую – виртуальную – реальность, свободную от ограничений вещно-физического мира, с новым типом связей, новыми возможностями целеполагания, моделирования будущей деятельности.

Начиная с эпохи Просвещения европейские мыслители ставили перед собой задачу не столько замены себя Мыслящими Машинами, сколько **возвращения** себе от этих Левиафанов функций принятия решений и построения жизни по своему разумению. Человек благодаря Духу, дарованному Богом, свободен от Природы, Физики, Математики, Числа. Его нельзя было исчислить, просчитать. Он не должен быть машиной или автоматом.

И вдруг на третьем тысячелетии от Рождества Христова в противовес этой точке зрения выясняется, что человек и продукты его мысли оцифровываются, исчисляются. Создается их цифровая копия, с которой можно проводить **любые** операции. Именно в машинном («искусственном») интеллекте вдруг увидели возможность освобождения от машин социальных.

Появилась каста жрецов (айтишники), которые могут создавать копии любых проявлений человеческой активности и индивидуальности (голоса, жеста, мысли). Могут создавать человеческих аватаров. Могут наделять их способностью принимать решения.

При этом создается иллюзия, что «цифровые технологии», включая технологии искусственного интеллекта, суть результат естественно-научной, т.е. в прямом смысле этих слов ЕСТЕСТВЕННОЙ и НАУЧНОЙ деятельности, аналогичной неумолимой силе законов природы, имеющей черты всеобщности, обязательной для всех (как, например, закон всемирного тяготения).

На самом деле процесс «оцифровки» всего и вся есть творчество ограниченной группы сочинителей кода, которые кладут в основу «сотворения мира» (цифрового) достаточно произвольные (и не единственные) решения по созданию языков программирования и последующего конструирования с их помощью цифровых пространств и населяющих их аватаров.

Сами того не осознавая, «архитекторы кодинга» вынуждены выстраивать собственные аналоги онтологий, гносеологий и диалектик, повторяя ошибки и заново решая проблемы, которые были поставлены и решены в истории философии.

Но в отличие от философов, которые редко становились «властителями дум» и почти никогда «властителями кошельков», IT-технологии изначально из мира науки перекочевали в мир масскульта и общества потребления. Цифровые решения изначально стали частью финансовых и медийных технологий.

И это не случайно. Вся современная социально-экономическая цивилизация, именуемая капиталистической, построена на... цифре. В ее основе лежит не качественно определенные труд, вдохновение или творчество, а **СТОИМОСТЬ** как основа **количественных** соотношений при добровольном обмене товарами между его производителями/собственниками. Количество. Цифра.

Не случайно, что цифровые технологии достаточно быстро освоили свое материнское лоно — сферу финансов. И даже покусались на самое святое — деньги, создав собственную, цифровую, сущность — криптовалюту. Полностью освобожденную от «первородного греха» вещиности.

Благодаря такому невероятному усилителю, как товарно-денежные отношения (рыночная экономика, когда сила не в правде, а в деньгах), в использование цифры включаются все, кто включен в товарно-денежные отношения = все члены современного общества. При этом побеждает то «цифровое решение», за которое голосует (рублем, долларом) масса. А масса любит те решения, которые она понимает, т.е. простые и яркие. Она не очень задумывается над тем, как это устроено.

Так незаметно интеллект небольшой группы «посвященных» начинает диктовать остальным членам рода человеческого моду на образ жизни и образ мыслей.

Выходом из данной ситуации, как ни странно это выглядит, может служить возвращение к истокам возникновения «цифровой цивилизации». А у истоков, связывающих «свободную» виртуальность IT-пророков с материальной реальностью, лежит математика **как порождение, «снятое» инобытие физического Мира**. Математика — это своеобразный мостик между объективным физическим миром с его ограничивающими полет виртуальной фантазии законами и «миром цифровым» с его «свободой», переходящей в произвол. «Оцифровка мира», оглядывающаяся на Математику, имеет шанс получить иммунитет от произвола и субъективности как в научном поиске, так и в социальных практиках, использующих продукты этого программирования.

Конкретным инструментам и техникам такого возвращения из «царства Свободы» в мир строгих правил и закономерностей и посвящена данная монография.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Монографии

1. *Лорьер Ж.-Л.* Системы искусственного интеллекта. М.: Мир, 1991.
2. *Саймон Х.* Нейронные сети: полный курс. М.: Издательский дом Вильямс, 2008.
3. *Шапиро Л., Стокман Дж.* Компьютерное зрение. М.: Бином. Лаборатория знаний, 2006.
4. *Шваб К.* Четвертая промышленная революция. М.: Эксмо, 2016.
5. *Ванник В.Н.* Восстановление зависимостей по эмпирическим данным. М.: Наука, 1979. 447 с.
6. *Казанцев В.С.* Задачи классификации и их программное обеспечение (пакет КВАЗАР). М.: Наука, 1990. 136 с.
7. *Мазуров В.Д.* Метод комитетов в задачах классификации и оптимизации. М.: Физматлит, 1990. 248 с.
8. *Мазуров В.Д.* Математические методы распознавания образов: учебное пособие / Урал. гос. ун-т им. А.М. Горького. 2-е изд., доп. и перераб. Екатеринбург: Изд-во Уральского ун-та, 2010. 101 с.
9. *Минский М., Пейпер С.* Перцептроны. М.: Мир, 1971. 262 с.
10. *Флах П.* Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных. М.: ДМК Пресс, 2015. 400 с.
11. *Педро Домингос.* Верховный алгоритм. 2016.
12. *Воронцов К.В.* Лекции по алгоритмам кластеризации и многомерного шкалирования. Курс лекций. МГУ, 2007.

Статьи и тезисы докладов

13. *Дудченко П.В.* Метрики оценки классификаторов в задачах медицинской диагностики // Молодежь и современные информационные технологии: Сборник трудов XVI Междунар. науч-практ. конф. Студентов, аспирантов и молодых ученых, 13 декабря 2018, Томск. Томск: Изд-во ТПУ, 2019. С. 164–165.
14. *Мак-Каллок У.С., Питтс В.* Логическое исчисление идей, относящихся к нервной активности. М.: Изд-во иностр. лит., 1956. С. 363–384.
15. *Горелик В.А., Трембачева О.С.* Решение задачи линейной регрессии с использованием методов матричной коррекции в метрике // Журн. вычисл. матем. и матем. физ. 2016. Т. 56, № 2. С. 202–207.

16. *Чемезов Д.С., Тюкачев Н.А.* Обзор основных методов классификации и кластеризации // Вестник ВГУ. 2009. № 2. С. 25–29.
17. *Старовойтов В.В., Голуб Ю.И.* Сравнительный анализ оценок качества бинарной классификации // Информатика. 2020. Т. 17, № 1. С. 87–101.
18. *Чернавин Ф.П.* Методический инструментарий оценки и моделирования кредитного риска по потребительским кредитам с применением комитетных конструкций: дис. канд. эк. наук: 08.00.13 / Чернавин Федор Павлович. Екатеринбург, 2016. 166 с.
19. *Кувшинов Б.М., Ширяев О.В., Шапошник И.И.* Система диагностики заболеваний методами распознавания образов и классификации в n-мерном пространстве / Б.М. Кувшинов, О.В. Ширяев, И.И. Шапошник // Информационные технологии. 2000. № 6. С. 43–47.
20. *Никонов О.И., Чернавин Ф.П.* Построение рейтинговых групп заемщиков физических лиц с применением метода комитетов / О.И. Никонов, Ф.П. Чернавин // Деньги и кредит. 2014. № 11. С. 52–54.
21. *Никонов О.И., Чернавин Ф.П., Медведева М.А.* Проблемы классификации: метод комитетов / О.И. Никонов, Ф.П. Чернавин, М.А. Медведева // Устойчивое развитие российских регионов: экономическая политика в условиях внешних и внутренних шоков: сборник материалов XII международной научно-практической конференции. Екатеринбург, 2015. С. 867–874.
22. *Чернавин Ф.П.* Применение метода комитетов для решения задач классификации / Ф.П. Чернавин. Изд-во Уральского Политехнического университета, 2018. С. 437–447.
23. *Чернавин Ф.П.* Применение нейронных сетей к задачам оценки вероятности дефолта по потребительским кредитам / Ф.П. Чернавин // Журнал научных публикаций аспирантов и докторантов. 2013. № 7. С. 21–25.
24. *Чернавин Ф.П.* Применение комитетных конструкций для принятия решений по потребительским кредитам / Ф.П. Чернавин // Экономика и предпринимательство. 2015. № 12. С. 143–149.
25. *Акбердина В.В., Чернавин Н.П., Чернавин Ф.П.* Построение рейтинговой модели оценки кредитного риска с применением метода комитетов / В.В. Акбердина, Н.П. Чернавин, Ф.П. Чернавин // Пермский финансовый журнал. 2017. № 2. С. 12–23.
26. *Никонов О.И., Чернавин Ф.П., Чернавин Н.П.* Применение метода комитетов к решению задач прогнозирования валютного рынка / О.И. Никонов, Ф.П. Чернавин, Н.П. Чернавин // Вестник УИЭУиП. 2015. № 2. С. 32–38.
27. *Чернавин Н.П., Чернавин Ф.П.* Применение метода комитетов к прогнозированию движения фондовых индексов / Н.П. Чернавин, Ф.П. Чернавин. М.: Изд-во ООО «Русальянс “Сова”». 2015. С. 307–320.
28. *Чернавин Н.П., Чернавин Ф.П.* Применение метода комитетов в техническом анализе инструментов финансовых рынков / Н.П. Чернавин, Ф.П. Чернавин // Современные научные исследования в сфере экономики: сборник результатов научных исследований. Киров: Изд-во МЦИТО, 2018. С. 1052–1062.
29. *Акбердина В.В., Чернавин Н.П., Чернавин Ф.П.* Применение метода комитетов к прогнозированию движения валютных курсов и цен на нефть / В.В. Акбердина, Н.П. Чернавин, Ф.П. Чернавин // Финансы и кредит. 2017. Т. 23, № 46. С. 2746–2762.

30. *Чернавин П.Ф., Чернавин Ф.П., Чернавин Н.П.* Сведение задач регрессии к задачам линейного целочисленного программирования // Анализ, моделирование, управление, развитие социально-экономических систем: сборник научных трудов XIV Всероссийской с международным участием школы-симпозиума АМУР-2020. Симферополь, 2020. С. 383–386.
31. *Гайнанов Д.Н., Чернавин Н.П., Чернавин П.Ф., Чернавин Ф.П., Рассказова В.А.* Выпуклые оболочки и выпукло отделимые множества в задаче многоклассового распознавания образов. Труды МАИ. 2019.
32. *Маккалох Дж., Питтс У.* Логические исчисления идей, относящихся к нервной деятельности. Автоматы. М.: ИЛ, 1956.
33. *Никонов О.И., Чернавин Ф.П.* Современные методы классификации. Метод комитетов / О.И. Никонов, Ф.П. Чернавин // Материалы конференции «Социально-экономическое развитие регионов России», 23 мая 2014.
34. *Никонов О.И., Чернавин Ф.П., Власов В.Е.* Сравнение метода комитетов и логит-модели в приложениях к решению задач прогнозирования валютного рынка / О.И. Никонов, Ф.П. Чернавин, В.Е. Власов // Материалы X Международной научной конференции по проблемам экономического развития в современном мире «Устойчивое развитие российских регионов: Россия и ВТО». Май 2013.
35. *Никонов О.И., Чернавин Ф.П.* Построение рейтинговых групп заемщиков – физических лиц с применением метода комитетов / О.И. Никонов, Ф.П. Чернавин // Деньги и Кредит. 2014. № 11. С. 52–55.
36. *Чернавин Ф.П.* Применение комитетных конструкций для принятия решений по потребительским кредитам / Ф.П. Чернавин // Экономика и предпринимательство. 2015. № 12–4(65–4). С. 143–149.
37. *Чернавин Ф.П.* Применение метода комитета большинства для принятия решения по выдаче кредита / Ф.П. Чернавин // Доклады всероссийской научной конференции АИСТ'13. Екатеринбург, 4–6 апреля 2013 г. С. 93–96.
38. *Никонов О.И.* Построение рейтинговых моделей с применением комитетных конструкций / О.И. Никонов, Ф.П. Чернавин, Н.П. Чернавин // Устойчивое развитие российских регионов: экономическая политика в условиях внешних и внутренних шоков: сборник материалов XII международной научно-практической конференции, Екатеринбург, 17–18 апреля 2015 г.
39. *Никонов О.И.* Проблемы классификации: метод комитетов / О.И. Никонов, Ф.П. Чернавин, М.А. Медведева // Устойчивое развитие российских регионов: экономическая политика в условиях внешних и внутренних шоков: сборник материалов XII международной научно-практической конференции, г. Екатеринбург, 17–18 апреля 2015 г.
40. *Чернавин Ф.П.* Методический инструментарий оценки и моделирования кредитного риска по потребительским кредитам с применением комитетных конструкций: дис. ... канд. экон. наук. 08.00.13. 2016.

Интернет-источники

41. *Греф Г.* Конгресс «Инновационная практика: Наука плюс Бизнес». <https://ria.ru/20161207/1483041883.html>
42. Новые тренды в сфере ИИ и машинного обучения в этом году. <https://www.crn/news/detail.php&ID=143082>

43. Толькова Т.Е., Владимирский М.А., Хабибуллина Н.Ф., Чернавин П.Ф., Чернавин Н.П. Способ определения активности специфического воспаления при наличии минимальных туберкулезных изменений у детей и подростков. Патент RU2728943 С1.

Иностранная литература

44. *Rumelhart D.E., Hinton G.E., Williams R.J.* Learning internal representation by error propagation // *Parallel Distributed Processing*. Cambridge, MA: MIT Press, 1986. Vol. 1. P. 318–362.
45. *Legendre A.M.* On Least Squares. New York: Columbia University, 2011.
46. *Lehmann E.L., Casella George.* Theory of point estimation. New York: Springer verlag, 1998.
47. *Willmott C.J., Matsuura Kenji.* Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance // *Climate Research*. 2005. Vol. 30. P. 79–82.
48. *Huber P.J.* Robust estimation of a location parameter // *Annals of Statistics*. 1964. Vol. 53(1). P. 73–101.
49. *Zou H., Hastie T.* Regularization and variable selection via the elastic net // *Journal of the Royal Statistical Society. Series B*. 2005. P. 301–320.
50. *Meindl B., Templ M.* Analysis of commercial and free and open source solvers for linear optimization problems. Institut f. Statistik u. Wahrscheinlichkeitstheorie, Austria, 2012.
51. CPLEX Optimization, Inc. (Incline Village, Nev.). 1995. Using the CPLEX base system: including CPLEX mixed integer solver and barrier solver options. CPLEX.
52. *Gkioulekas Y., Papageorgiou L.G.* Piecewise regression through the Akaike information criterion using mathematical programming // *IFAC PapersOn-Line*. 2018. Vol. 51(15). P. 730–735.
53. *Wang B.Q., Chukova S., Lai C.D.* On the relationship between regression analysis and mathematical programming // *Journal of applied mathematics and decision sciences*. 2004. Vol. 8(2). P. 131–140.
54. *Yang L., Liu S., Tsoka S., Papageorgiou L.G.* A regression tree approach using mathematical programming // *Expert Systems With Applications*. 2017. Vol. 78. P. 347–357.
55. *Zhuravlev Y.I.* On the algebraic approach to solving problems of recognition and classification // *Problems of cybernetics*. Moscow: Nauka, 1978. Vol. 33. P. 5–68.
56. *Haixiang G., Shang J., Mingyun G., Yuanyue H., Bing G.* Learning from class-imbalanced data: Review of methods and applications // *Expert Systems with Applications*. 2017. Vol. 73. P. 220–239.
57. *Choi S.S., Cha S.H., Tappert C.C.* A survey of binary similarity and distance measures // *Journal of Systemics, Cybernetics and Informatics*. 2010. Vol. 8(1). P. 43–48.
58. *Canbek G., Sagioglu S., Temizel T.T., Baykal N.* Binary classification performance measures/metrics: A comprehensive visualized roadmap to gain new insights. International Conference on Computer Science and Engineering, Antalya, Turkey, 5–8 October 2017. Antalya, 2017. P. 821–826.
59. *Sokolova M., Lapalme G.* A systematic analysis of performance measures for classification tasks // *Information Processing & Management*. 2009. Vol. 45, no. 4. P. 427–437.

60. *Valverde-Albacete F.J., Pelbez-Moreno C.* 100% classification accuracy considered harmful: the normalized information transfer factor explains the accuracy paradox // *PLoS One*. 2014. Vol. 9(1). 10 p.
61. *Powers D.M.* What the F-measure doesn't measure: Features, Flaws, Fallacies and Fixes. 2015.
62. *Fawcett T.* An introduction to ROC analysis // *Pattern Recognition Letters*. 2006. Vol. 27, no. 8. P. 861–874.
63. *Wei J.M., Yuan X.J., Hu Q.H., Wang S.Q.* A novel measure for evaluating classifiers // *Expert Systems with Applications*. 2010. Vol. 37, no. 5. P. 3799–3809.
64. *Davis J., Goadrich M.* The relationship between Precision-Recall and ROC curves // *Proceedings of the 23rd International Conference on Machine Learning*, 25–29 June 2006, Pittsburgh, Pennsylvania, USA. Pittsburgh, 2006. P. 233–240.
65. *Boughorbel S., Jarray F., El-Anbari M.* Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric // *PloS One*. 2017. Vol. 12(6). <https://doi.org/10.1371/journal.pone.0177678>
66. *Indira Rustempasic, Mehmet Can.* Diagnosis of Parkinson's Disease using Principal Component Analysis and Boosting Committee Machines // *South-east Europe journal of soft computing*. 2013. Vol. 2, no. 1. P. 102–109.
67. *Karaduzovic-Hadziabdica K., Kokerb R.* Diagnosis of heart disease using a committee machine neural network // *Proceedings of the 9th International Conference on Applied Informatics*. Eger. Hungary. 2014. Vol. 1. P. 351–360.
68. *Lee T., Richards J.A.* Piecewise linear classification using seniority logic committee methods, with application to remote sensing // *Pattern Recognition*. 1984. Vol. 17, no. 4. P. 453–464.
69. *Tang Ho-Man, Lyu M.R., King I.* Face recognition committee machines: dynamic vs. static structures // *12th International Conference on Image Analysis and Processing*. 2003. Available at: <https://ieeexplore.ieee.org/document/1234037> (Accessed 13 August 2019).
70. *Nikonov O.I., Medvedeva M.A., Chernavin F.P.* Using the Committee Machine Method to Forecasting on the FOREX // *IEEE2015 Second International Conference on Mathematics and Computers in Sciences and in Industry (MCSI)*. 2015. P. 240–243.
71. *Akberdina V.V., Chernavin N.P., Chernavin F.P.* Application of the committee machine method to forecast the movement of exchange rates and oil prices / В.В. Акбердина, Н.П. Чернавин, Ф.П. Чернавин // *Дайджест-Финансы*. 2018. Т. 23, № 1. С. 108–120.
72. *Akberdina V.V., Chernavin N.P., Chernavin F.P.* Application of the Committee Machine Method to Forecast an Increase in the USD/RUB Exchange Rate Volatility // *Proceedings of the 5th International Young Scientists Conference on Information Technologies, Telecommunications and Control Systems*. Yekaterinburg, Russia, 2018. Available at: <http://ceur-ws.org/Vol-2298/paper19.pdf> (Accessed 20 August 2019)
73. *Gainanov D.N., Chernavin P.F., Rasscazova V.A., Chernavin N.P.* Convex hulls in solving multiclass pattern recognition problem // *Learning and Intelligent Optimization*. 14th Intern. Conf., LION14, Athens, Greece, May 24–28, 2020. P. 390–401.
74. *Pardalos P.M., Li Y., Hager W.W.* Linear Programming Approaches to the Convex Hull Problem in Rm // *Computers Mathematics Applications*. 1995. Vol. 29, no. 7. P. 23–29.

Научное издание

**Чернавин Павел Федорович,
Гайнанов Дамир Насибуллович,
Панкращенко Виктор Николаевич,
Чернавин Федор Павлович,
Чернавин Николай Павлович**

**МАШИННОЕ ОБУЧЕНИЕ
НА ОСНОВЕ ЗАДАЧ
МАТЕМАТИЧЕСКОГО
ПРОГРАММИРОВАНИЯ**

Редактор *Л.В. Филиппова*

Художник *В.Ю. Яковлев*

Корректоры

А.Ю. Обод, С.О. Розанова

Подписано к печати 25.11.2021

Формат 60 × 90¹/₁₆. Гарнитура Newton

Печать офсетная

Усл.печ.л. 8,0. Уч.-изд.л. 10,0

Тип. зак.

ФГУП Издательство «Наука»

117997, Москва, Профсоюзная ул., 90

E-mail: info@naukaran.com

naukapublishers.ru

ФГУП Издательство «Наука»

(Типография «Наука»)

121099, Москва, Шубинский пер., 6